ARI Research Note 90-62

Distributed Problem Solving: Adaptive Networks With a Computer Intermediary Resource

John Lyman, Steven White, and Paul E. Young
University of California

for

Contracting Officer's Representative
Michael Drillings

Basic Research
Michael Kaplan, Director

July 1990

FIFTY YEARS OF SERVICE
1940 - 1990

United States Army
Research Institute for the Behavioral and Social Sciences

Approved for public release; distribution is unlimited

U.S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency Under the Jurisdiction of the Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON Technical Director	JON W. BLADES COL, IN Commanding

---

Research accomplished under contract for
the Department of the Army

University of California

Technical review by

Michael Drillings

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS --
2a. SECURITY CLASSIFICATION AUTHORITY -- 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE --	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
4. PERFORMING ORGANIZATION REPORT NUMBER(S) --	5. MONITORING ORGANIZATION REPORT NUMBER(S) ARI Research Note 90-62
6a. NAME OF PERFORMING ORGANIZATION Department of Mathematics, Science, and Engineering, University of California at Los Angeles 6b. OFFICE SYMBOL (If applicable) --	7a. NAME OF MONITORING ORGANIZATION U.S. Army Research Institute Office of Basic Research
6c. ADDRESS (City, State, and ZIP Code) Los Angeles, CA 90024	7b. ADDRESS (City, State, and ZIP Code) 5001 Eisenhower Avenue Alexandria, VA 22333-5600
8a. NAME OF FUNDING / SPONSORING ORGANIZATION U.S. Army Research Institute for the Behavioral and Social Sciences 8b. OFFICE SYMBOL (If applicable) PERI-BR	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MDA903-84-C-0355
8c. ADDRESS (City, State, and ZIP Code) Office of Basic Research 5001 Eisenhower Avenue Alexandria, VA 22333-5600	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO.: 61102B PROJECT NO.: 74F TASK NO.: n/a WORK UNIT ACCESSION NO.: n/a

11. TITLE (Include Security Classification)
Distributed Problem Solving: Adapti^ve Networks With a Computer Intermediary Resource

12. PERSONAL AUTHOR(S)
Lyman, John; White, Steven; and Young, Paul

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim	FROM 84/10 TO 88/06	1990, July	95

16. SUPPLEMENTARY NOTATION Michael Drillings, contracting officer's representative

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Problem solving; Network; Human factors; Human-computer interaction

19. ABSTRACT (Continue on reverse if necessary and identify by block number)
Distributed problem solving networks are omnipresent decision-making structures with often untapped potential. The power of these networks cannot fully be recognized until the basic processes within the network are understood. A major source of disruptions in the network is the degradation of processes caused by uncertainty. Many uncertainty sources have technological solutions, such as increased sensor capability, increased communication bandwidth, or increased processing power. Inter-agent uncertainty is a more fundamental group dynamics problem without an obvious technological solution. Establishing and understanding inter-agent uncertainty reduction processes is necessary for a more effective utilization of the distributed problem-solving network. Forcing disagreeing agents to resolve their differences under control of a predefined set of rules is a proposed method of attacking this problem. The first report proposes a testbed design to study the process using input/output analysis. The difference resolving rules are inputs; the measures of group performance, the outputs. The second focus is the assessment programs/instructions.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Michael Drillings	22b. TELEPHONE (Include Area Code) (202) 274-8722 22c. OFFICE SYMBOL PERI-BR

DD Form 1473, JUN 86 Previous editions are obsolete.

DISTRIBUTED PROBLEM SOLVING: ADAPTIVE NETWORKS WITH A COMPUTER INTERMEDIARY RESOURCE

CONTENTS

	Page
INTRODUCTION AND OVERVIEW	1
DISTRIBUTED PROBLEM SOLVING	2
HYPOTHETICAL TESTBED	19
Introduction	19
Required Testbed Characteristics	19
Testbed Scenario Generator	20
DESIGN FEATURES OF THE HYPOTHETICAL TESTBED	24
Introduction	24
Problem Definition and Goal Setting	26
Message Categorization	39
Difference Resolution	50
System Activation	52
Intelligent Intermediary	54
DATA COLLECTION USING THE HYPOTHETICAL TESTBED	62
CONCLUSIONS	66
REFERENCES	69

LIST OF FIGURES

Figure 1.	Four possible structural configurations for distributed networks	5
2.	Evolution of a multiple agent network	9
3.	Group dynamics of a multiple agent network	17
4.	Partial relational flow of the limited resources testbed	21
5.	Testbed timeline	23
6.	Feedback loops in the network	25

CONTENTS (Continued)

		Page
Figure 7.	Simple representation of the problem solving process	26
8.	Goal setting	31
9.	Sample output of the utility assessment process	33
10.	Functional depiction of an agent's workspace	42
11.	Data of message types sent by agents during one decision-making time period	45
12.	Functional representation of the message sending window . .	47
13.	Message sending in a multiple agent network	49
14.	Agents' message evaluation data	50
15.	Multiple agent different resolving with time sequencing effects	52
16.	Collectible message evaluation and difference information .	53
17.	Predicted relationship between problem complexity, system overhead costs, and performance	55
18.	Functional representation of the intelligent intermediary .	57
19.	Data collected by the system available for use by the intelligent intermediary	58

LIST OF SYMBOLS

S -- summation

DG_{ij} -- vector of differences in goal assignment to ith message from jth agent

DE_{ij} -- differences in message evaluation between ith and jth agent

DP_{ij} -- vector of differences in priority assignment to ith message from jth agent

a_i -- ith alternative solution

A_{ij} -- jth resource allocation alternative at time i

c(m) -- lower critical threshold level of mth resource

D_i -- decision made at time i

f(arg) -- function of the argument

G -- overall group goal

G_{ijk} -- goal assessment by the kth agent of the ith message from jth agent

G_{ijkF} -- final goal assessment by the kth agent of the ith message from jth agent

g_i -- ith group sub-goal

m(c) -- measure of problem complexity

M_j -- message from jth agent

M_{ij} -- ith message from jth agent

p(k) -- probability of kth state of nature occurring

q(m) -- the physical measure of mth resource

q_{im} -- the physical measure of mth resource at time i

P_{ijk} -- priority assessment by the kth agent of the ith message from jth agent

P_{ijF} -- final priority assessment by the kth agent of the ith message from jth agent

PI_i -- performance index at time i

R_{ijk} -- resultant bundle of resources from choosing the jth resource allocation alternative at time i given the kth state of nature occurs

s(m) -- upper saturation level of the mth resource

t(c) -- threshold level of problem complexity

U_{im} -- the utility value of the mth resource at time i

W_{im} -- the importance weighing of the mth resource at time i

DISTRIBUTED PROBLEM SOLVING: ADAPTIVE NETWORKS WITH A COMPUTER INTERMEDIARY RESOURCE

1.0 INTRODUCTION AND OVERVIEW

A functional design of a testbed for use in the investigation of human interactions in a distributed problem solving network is proposed in this report. Before describing the testbed design features, issues that must be resolved before the benefits of the distributed problem solving network exceed its costs will be reviewed. This will be followed by a discription of specific testbed features that have been designed to address specific problem areas. The report closes with a discussion of the types of data that can be collected in experiments that use the proposed testbed. The data selection chosen is designed to help understand the underlying processes occurring in distributed problem solving networks. The main process of interest is the reduction of inter-agent uncertainty as a function of difference resolution rules. The rules are used to drive a resolution of the agents' disagreement on information evaluation towards consensus.

2.0 DISTRIBUTED PROBLEM SOLVING

Our emphasis in this report is on the situation in which spatial separation requires a distributed communications network to handle the requirements. of a task. A battlefield scenario with various field units relaying information to a command and control center would be an example of distributed problem solving. More specifically, it would be an example of distributed sensing in a situation assessment problem. Any large scale technical project, such as the Apollo or Space Shuttle program, could serve as an example of such a problem. Typically, the problem to be solved may be so complex and wide ranging as to require expertise and information from many knowledge disciplines. In such problems no one agent can solve the problem. For solution, the problem must be decomposed into smaller local tasks that are distributed within a group of agents. The cooperative efforts of the group must also be centrally coordinated in order to solve the problem. For the sake of a common starting point, this report will adopt the definition of distributed problem solving proposed by Smith and Davis (1981), which is as follows:

> Distributed problem solving is the cooperative solution of problems by a decentralized and loosely coupled collection of knowledge sources (KS's) (procedures, set of rules, etc.), located in a number of distinct processor nodes. The KS's cooperate in the sense that no one of them has sufficient information to solve the entire problem; mutual sharing of information is necessary to allow the group as a whole to produce an answer. By decentralized we mean that both the control and data are logically and often geographically distributed; there is neither global control nor global data storage. Loosely coupled means that the individual KS's spend the great percentage of their time in computation rather than communication.

In his book on man-machine systems, Parsons (1972) described the case histories of two of the earlier research groups that investigated group problem solving. The experiences of the two research groups turned out to be valuable lessons in group problem solving and the observations that Parsons made about the two groups point out the complexities of group problem solving.

The first group worked on Project Cadillac, which lasted from about 1948 to 1955. Parsons described their experience as follows:

> Possibly one reason why it had been difficult to take this step earlier was the relative unfamiliarity - largely unavoidable - of virtually all the research people, at all levels, with man-machine systems, AEW operations, and multivariable experiments. In addition, it became clear that staffing and leadership of this kind of complex, large-scale research inevitably presents challenges. Inexperience and lack of well-established guidelines seem unlikely to promote humility or harmony, especially if the professional linkage is tenuous between research managers and researchers, when they come from diverse disciplines with differing approaches to the task. Interpersonal discord within the project was accompanied by a considerable turnover of personnel, who, when they achieved some degree of sophistication, did not necessarily remain long on the scene.

Parsons points out one of the more important aspect of this set of experiments was that it served as valuable training ground for a new domain of applied science, system learning processes.

The second group, including a former investigator form Project Cadillac, worked on a series of four experiments (Casey, Cowboy, Cobra, and Cogwheel) from 1952 to 1954. Parsons described their experiences as follows:

> How did it all start? In 1950 a number of psychologists attended a summer conference which RAND had called because its engineers and scientists were uncertain how to assess the contribution of human operators to the effectiveness - and degradation - of the future systems which they were studying for the Air Force. One of the attendees, Kennedy, had been heading a program at Tufts University for collecting human engineering data and had participated in the Applied Psychology Panel's development of applied research in World War II. After accepting an invitation to join Rand in 1951, he brought two other psychologists, Chapman and Biel, to the RAND corporation. Chapman had been directing the technical program at Project Cadillac and thereby had acquired know-how for creating a simulation laboratory and conducting complex experiments. Biel, whose experiences during World War II was likewise pertinent, came from human engineering research in the Aero-Medical Laboratory at Wright Field. The team was increased to four by the addition of Newell, a RAND physicist and mathematician who had been working on Air Force logistics problems.
>
> The somewhat diverse composition of this team was important to its success, as each member contributed special talents, yet they were united in the spirit of scientific discovery. Kennedy communicated with the RAND management as the spokesman, but the four members interacted with each other as peers, and

plans and courses of action were usually based on consensus, often following considerable discussion.

Both groups consisted of individuals with assorted skills brought together to work towards a common goal. However, the experiences of the two groups differed appreciably. Parsons suggests the second group had a better understanding of what was to be done and based on the previous experience had a better feel for what had to be done, leaving it without some of the problems that affected the performance of the first group. The question of motivation and self interest was also raised. The working structure of the first group caused friction within the group, probably lowering morale and motivation. Also as some researchers in the first group reached a certain level of experience, they left the group for various reasons. Personnel turnover in the first group was an important factor that was believed to contribute heavility to its lowered group performance.

Other concerns included questions of lost system capability (whether through graceful degradation or by the outright loss or destruction of a part of the network) and counter measures employed by the opposition in adversarial situations. Although these were concerns for systems in the real world, they introduced complex issues that range beyond the scope of this report.

In the proposed experimental testbed, the problem would be well defined, the group goals would be identified in advance, and guidelines as how to interpret the problem, goals, and tasks would be provided to the group. The structure of the group can influence group performance and as such the structure would be defined in advance and fixed through any experiments. Figure 1 shows four common network structures. The network structure given emphasis for this report is the star structure which accords each individual equal status within the group.

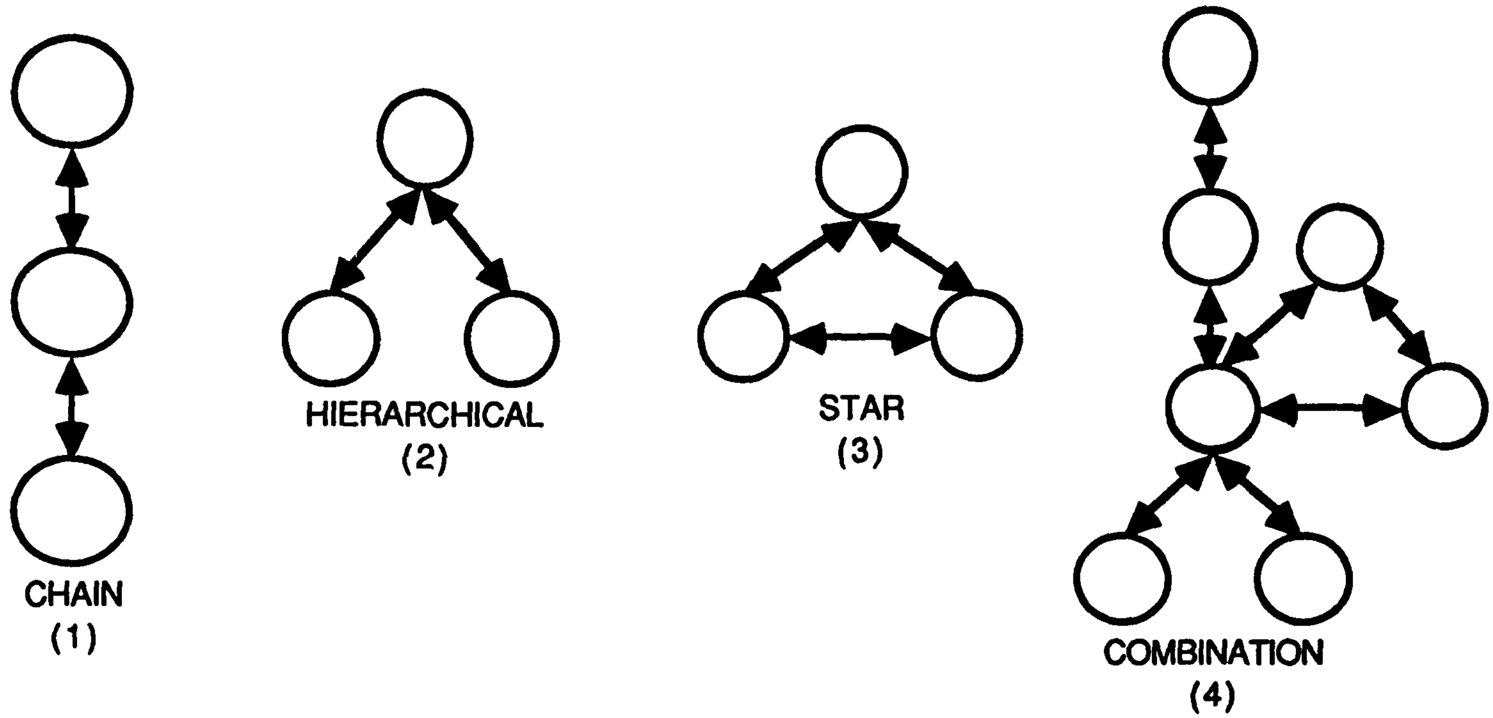


FIGURE 1 - FOUR POSSIBLE STRUCTURAL CONFIGURATIONS FOR DISTRIBUTED NETWORKS

This report will, in part, follow in the footsteps of Project Cadillac. Parsons described one aspect of Project Cadillac as follows:

> ... independent variables were not systematically incorporated because the experimenters wished to derive concepts or hypotheses from what happened in the laboratory, rather than test preconceived hypotheses. Their only assumption was that a crew would learn how to function. They wanted to see how the organization under scrutiny, as a self organizing one, would organize itself procedurally - not structurally.As an indication of the generality which the researchers wished to ascribe to their model, they initially referred to it as the information processing center (IPC).

This report is concerned with how the group goes about solving problems (procedures), how their problem solving procedures change, and how to accelerate group acquisition of more effective problem solving procedures. Though hypotheses can be generated as to what will happen to group performance when feedback control parameters are introduced and varied, it is conceded a more sound and rigorous rationale could be derived if results from implementing the proposed system were available for analysis.

Issues such as self-interest, motivation, learning, or attitudes that influence how a group performs are the more basic areas to be investigated. The following excerpts from Parsons shows the importance of some of these factors. The first excerpt is as follows:

> What general concepts came out of the RAND SRL experiments? The experimenters believed that they had proved their basic, single hypothesis or assumption, that a motivated organization with a goal can and will adapt when it faces new situations and problems; it will solve its own problems. But why and how does it do so? According to Chapman et al:
>
> "The members of each crew became an integral unit in which many interdependencies and coordinating skills developed. And each crew learned to perform more effectively. This learning showed itself in procedural short cuts, reassignment of functions, and increased motor skills to do the job faster and more accurately.
>
> We believe that 'debriefings' following each session, where the operating results were reviewed, were crucial to the learning that lead to improved performance. But we have been unable to relate the content of these discussions directly to crew development. Procedures were frequently changed without any sign that an operating problem had been recognized or a

solution proposed. As a matter of fact, procedural changes moved in one direction while discussions went in another.

A second excerpt is as follows:

> Expressions of attitudes by members of the crew were noted by coding "attitude" cards. Although the experimenters had considered a method of distinguishing between positive and negative attitudes and of indicating by whom, to whom, and about what the attitudes were expressed, only the fact that an attitude had been expressed was coded during the experiment. After the experiment the crew received a sociometric questionnaire, an attitude questionnaire, and one concerned with procedures; and several psychodrama sessions were conducted with some of the subjects.

Another reason for investigating only the more basic aspects of group problem solving is to avoid becoming a slave to data collection or losing ideas in a sea of data. By starting with the more simple issues, it is likely less data will be needed and these data will probably have less likelihood of being confounded by higher order phenomena. Parsons offered the following observation from an experimenter to point out the problems of data collection and reduction:

> Voice records were obtained for that experiment but remained unexamined. "we collected as much data about the crews and their behavior as we could because we were searching for a framework rather than testing a hypothesis," the experimenters commented. "Only part of the data has been successfully coded or explored at any length although literally hundreds of pretty hypotheses have been lost in it. Although much of the data has been used only to explain specific incidents, it should prove of more general value once we know the appropriate questions to ask of it."

The particular area of interest of the present report is the class of distributed problem solving circumstances involving humans, with or without local computer aiding, in a spatially distributed network, cooperating to solve problems. This has become and will continue to be an important decisionmaking structure. Man is the knowledge source. Individual men or women will be referred to as agents. The term agent is preferred over the term expert since it is less specific in the assumption of an individual's experience base or expertise. The term , "expert", also begins to overlap into the field of expert systems and might cause some confusion.

There are three basic reasons for assuming no one agent can solve the global problem. First, the agent seldom has access to all the relevant information and observations necessary to solve the problem. Second even with only partial information, the information processing load is usually too great for one agent and Simon's concept of bounded rationality will come into play. The last reason is the fact that a single agent will probably lack sufficient knowledge in all the problem domains necessary to interpret the information and then solve the global problem. The individual agent will suffer from the combined effects of stress due to work load and stress due to situation uncertainty. A multiple agent system is thus needed. The most common strategy used when deploying a multiple agent system is to decompose the global problem into local sub-tasks. Each subtask is then assigned or taken by the agent most suited to handle it. During the problem solving process any agent can query the network for information held or observations made by the other agents. Each agent arrives at a solution for his individual sub-task or sub-tasks. Next, the group somehow links these lower level solutions together in order to arrive at a solution to the global problem. On the surface, the advantages of the multiple agent system are that the agents can act as distributed sensors and gather information for the entire network and the work load is distributed throughout the network by means of problem decomposition. The network can also have as many agents as necessary to cover the required problem domains. Figure 2 gives a simple representation showing the advantages of a multiple agent system over a single agent system. The knowledge base of any one agent, Agent 1 for example, may be only a small part of the knowledge needed to solve a problem. Thus, by pooling their knowledge, the agents may produce a better solution then the solution offered by any one of them working alone. An agent can conceivably increase his ability to solve problems by observing the problem solving algorithms or heuristics of others in the system. He can use that knowledge to enhance or refine his own problem solving tools or he can adopt a new technique and adapt it for his own purposes.

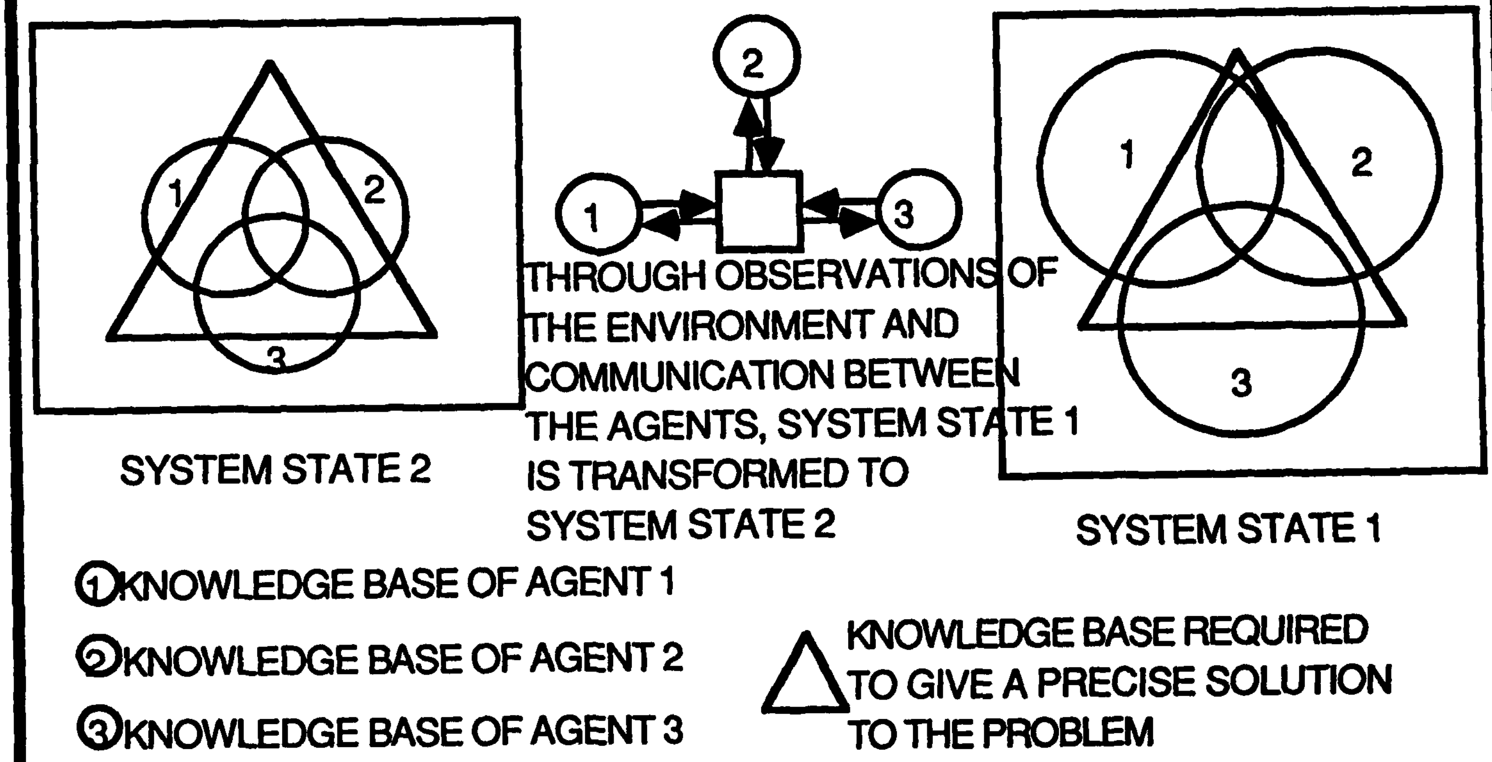


FIGURE 2 - EVOLUTION OF A MULTIPLE AGENT NETWORK

Finally by observing the environment and by communicating with one another, the agents can reduce the areas of uncertainty and further increase their problem solving capabilities. Uncertainty of the problem environment may be reduced to the extent that agents increase their ability to correctly interpret observations of the problem environment. Further reduction is achieved as agents learn to relay information to those most suited to interpret or use each particular piece of it. The cliches that one learns early in life, viz. things are not a simple as they first seem, and one seldom receives something for nothing are especially pertinent for group learning processes.

The multiple agent system only lessens the problems that were to be eliminated. The processing load of each agent in a multiple agent network may be similar to the processing load experienced by the agent in the one agent system. If the objective of the multiple agent system is to simply reduce the work load stress experienced by an agent, then establishing a multiple agent system to handle the same class of problem and to yield solutions similar in quality as the single agent system will probably fulfill the above objective. However, this implementation of a multiple agent system seems to be an expensive and wasteful use of resources that ignores the full potential of such a system.

Similarly, if we wish to extend the multiple agent system to the limits of its capabilities in order to reach better solutions, the agents in the network may experience load stress levels similar to that experienced by the agent in the single agent system. Whether the agent is working as an individual or in a problem solving team, a sufficiently complex problem will cause the agent's capabilities to become saturated. The difference between the two problem solving structures is the multiple agent system has potential to process more information. The quantity of information being processed can be increased. The types of information processed can be increased. Also the information can be processed at a finer level of detail. While the global problem is decomposed into tasks that are distributed in the network, the work load stress is not necessarily lowered in the multiple agent system. Rather the quantity and quality of the

information processing can be increased, the class of solvable problems can be expanded, and better solutions can be offered.

Each agent represents a potential increase in the information processing capability of the network. This fact is important since each agent may also act as sensor for the network. Any increased sensory capability that results means more information can be collected and made available to the network. Therefore an increase in the ability to process information may be needed to make use of the additional information. Even with the increased sensory capability, the entire problem environment can not necessarily be observed. Some significant events may remain undetected and cause uncertainty to exist, but the number of undetected events can ordinarily be lowered and the uncertainty reduced.

Uncertainty may be further reduced by introducing agents with different knowledge bases if they cover a larger part of the problem environment. While each agent can represent a knowledge source in a different area of the problem environment and increase the total knowledge base of the network, each agent does not necessarily have complete knowledge of the problem domain in which he specializes. Some unknown areas of knowledge may continue to exist, injecting uncertainty into the problem solving process, but at least the unknown areas of knowledge can be reduced.

Finding ways to reduce uncertainty in a distributed network is one of the main goals of this report. Uncertainty reduction is the fundamental concept behind the design of the proposed testbed. The amount of uncertainty present in the problem solving process will directly influence the amount of information and communication required to reduce the uncertainty to make the problem solvable by the agents with some acceptable level of confidence. Yang, et al (1985) support this contention when they state that incomplete and inexact knowledge or uncertainty about the problem environment and problem domains will lead to generating a number of competing hypotheses and associated tasks that must be communicated and coordinated.

Reduction of uncertainty is a key design criterion. They suggest the following list of common sense concerns a designer of a distributed system should consider:

- appropriate distribution of subproblems;
- appropriate control mechanisms to maintain global coherence, to assure efficient use of knowledge sources, and to achieve optimal performance;
- communication policies to allow efficient exchange of information between processing nodes.

One could always attempt to increase the knowledge base, sensory capability, and processing capability of the network by adding agents. However the costs of such a move may add up quickly and diminish the marginal rate of return of this option. A point may be reached where the addition of an agent may cause the overall network performance to drop. In the worst case, the network might pass some critical threshold and lock-up or shut down completely. This is analogous to traffic gridlock experienced in many major cities. Either the load must be reduced (reduce number of cars on the road), or the number of or capacity of the transmission lines must be increased (increase available roadways), or the procedures and protocols (traffic laws and driving habits) must be adjusted in order to alleviate the problem. The designer of a multiple agent system must carefully analyze how adding an agent, replacing one agent with another, or even removing an agent will affect the gains realized by it. This last caveat aside, Lesser and Corkill (1981) point out the advantages that may accrue in a multiple agent system that do not exist in a single agent network, in addition to the three mentioned above, are:

- increased reliability and flexibility;
- enhanced real-time response;
- lower communication costs;
- lower processing costs;
- reduced software complexity.

Though this is far from an exhaustive list, the promise of such gains would justify investigation of a distributed system if its costs can be justified. Offsetting the advantages of the multiple agent system are any control, coordination and communication problems introduced.

Control is achieved when an agent's actions accord with the group's goals. In the course of the problem solving process an agent will encounter occasions when he must choose one from a group of alternatives. *Mechanisms for picking the "correct"* alternative are referred to as control mechanisms. In certain distributed systems, control of the network is distributed. Control can be discussed at two levels. The first level is how to make an agent want to act in accordance with the rest of the network. Regardless of any formal policies or hierarchical structure that may have been established, each agent by virtue of his physical isolation, or the power bestowed on him by his unique area of expertise can operate with some degree of autonomy, thus potentially disrupting the flow and control of the network. Agents that do not intentionally subvert the group's efforts or wishes will be referred to as cooperative *agents.*

The second level of control is maintaining a cooperative agents' efforts in line with the group's goals. If the goals are being met, the agent can continue working as in the past. If the goals are not being met, the agent must consider corrective action. In either case, the agent has to receive some indication of how well he is meeting the group's goals in order to evaluate his performance. Some standard and measure of performance must be defined and made known to the agents in the network.

Even if the agents are under control in the both senses mentioned above, coordination of the individual agent's efforts is another cost of a distributed network. Assuming the problem can be meaningfully decomposed or partitioned, coordination in a network is achieved by mechanisms used to define and decompose the problem, assign tasks to the agents, synthesize a solution from partial solutions, and implement the final solution to the problem. The coordination tasks in a network increases the communication load and complexity. Some cost is incurred regardless of the amount of uncertainty in the problem solving environment. Uncertainty magnifies this problem, which grows as the number of agents in the system increases.The control and coordination problem was recognized by Smith and Davis (1981). They characterize cooperative distributed problem solving network as either task-sharing or result-sharing. When a problem can be decomposed into independent kernels (a nondecomposable sub-problem) and the solutions to the kernels can be linked into a solution of the overall problem, a task-sharing network can be used advantageously. Result-sharing is used when a global problem cannot be nicely decomposed. The agents have unique perspectives within the network and their partial results must be continually shared in order to solve the overall problem.Smith and Davis skipped over the problem decomposition phase of the problem solving process and started at the point where each agent can obtain one or more kernels to solve on his own. The group then gathers up the efforts of the individual agents and synthesizes a solution to the global problem. The fundamental question Smith and Davis addressed is how to distribute the tasks. They suggest the metaphor of negotiations and contracts to distribute the tasks and to coordinate and control the agents in the network. This so-called contract net scheme begins by assigning a manager to each kernel. The manager is responsible for finding solutions to his kernels. He may choose to do the work himself or call for other agents' bids to solve his kernels. Responding agents are referred to as contractors. The manager looks over the qualifications of the competing

contractors and then chooses a final contractor to execute the solution of each kernel. The contract net thus distributes the workload in the network to the most capable agents available at the time. It assigns responsibility to individuals in the network by identifying them as managers of specific kernels, thus achieving some level of control and coordination. The contract net does not force the group into a single problem solving hierarchy, but allows sub-groups to address problems as they arise. The contract net also allows mechanisms for communication and coordination to be established locally instead of at the global level. Result sharing is characteristic of a class of problems to which the contract net is not easily applicable. An example is a scene assessment problem where the agents have partially overlapping views of the scene to be interpreted. An iterative problem solving process is required in which the agents must constantly exchange partial solutions or guesses and refine them in order to arrive at a final solution. Areas of responsibility are not easily separable. Smith and Davis offer no explicit mechanism for approaching result-sharing dominated problems, and have only scratched the surface of the coordination problem. The communication problem entails getting the correct message content to the correct end user of the information in a timely manner. Communication between the agents can be investigated by installing an on-line message categorization system. This system will establish new feedback loops, the basic mechanism used to reduce uncertainty and to improve performance in the distributed network. These loops will allow the agents to compare their assessments of the information as it passes through the network. By having to interact with others to resolve any differences, each agent learns about himself and the other agents in the system. They expand their experience base by exchanging the past experiences that led to any competing

interpretations. Borrowing the Johari Window Model of Luft and Ingham for group dynamics described [Luft (1970)], the learning experience of the agents can be as depicted in Figure 3. Resolving differences not only allows the agents to learn about each other, but yields better interpretations of messages and decisions. Also, as the system establishes a database about each agent as a part of its overall database, the system can start functioning as an intelligent intermediary and help improve communications between agents. One might also expect that the anxiety or stress level to be reduced somewhat since the final outcome is arrived at in a state of mutual agreement and support. The final interpretation may be a totally new interpretation, one that was overlooked, a synthesis of the original interpretations, or a choice from the original interpretations. Whatever the final outcome may be, the resolution of any differences allows each agent on average to use the information more effectively for making better decisions, reducing the number of competing hypotheses and accompanying tasks, and establishing the potential for more streamlined communications in the future as each learns about the other in the system. Reducing this "agent uncertainty" is one of the main objects of this report.

Another intent of this report is to propose a specific vehicle for gathering data about the more fundamental aspects of processes in distributed networks. Along the way, some conjectures will be offered on how to improve coordination, communications, and control in a distributed network, whether the problem is task-sharing, result sharing, or both. The essence of the problem is that, in addition to the data and knowledge sources, data acquisition, and data processing being distributed among autonomous agents in the network, the planning, decisionmaking, control and implementation of decisions are also distributed among agents separated by a sea of uncertainty. A system that provides overall control and coordination among agents must be devised if the promise of the distributed network is to be realized. Questions

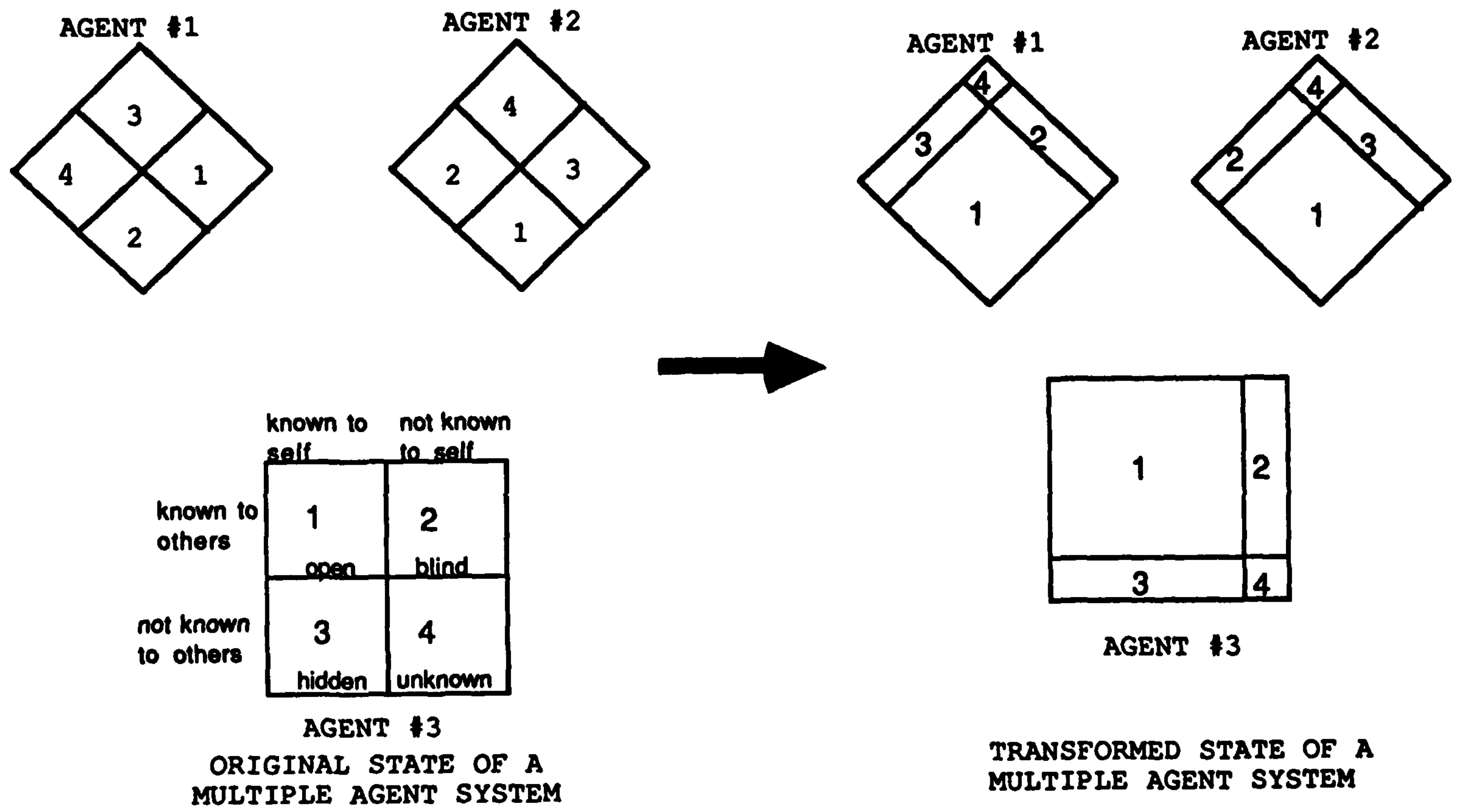


FIGURE 3 - GROUP DYNAMICS OF A MULTIPLE AGENT NETWORK

about sensors, data gathering and processing, hardware requirements and costs, software protocols and complexities, etc. remain. They represent real world constraints, but this report concentrates on proposing a specific arrangement for investigating ideas or methods of increasing the reliability, flexibility, and real time performance of the network. This will be done by trying to solve the communication, control, and coordination problems inherent within a distributed system. A key element for success will be the ability to reduce uncertainty in the system. In particular, the unknowns about each other that cause the agents to engage in ineffectual behavior or efforts. Before proceeding to a discussion of the issues a top level description of a hypothetical testbed will be given. This testbed will serve as a source of examples used to clarify or illustrate points made during the discussion. Discussion of the specific communication, control, and coordination mechanisms will also be developed.

3.0 HYPOTHETICAL TESTBED

3.1 Introduction

In order to experimentally test some of the ideas that will be put forth later in this report, a distributed problem solving testbed would be useful. While the actual implementation of a testbed is beyond the scope of this work, a paper design of a hypothetical testbed may help clarify or illustrate points that will be made in the discussion to follow

3.2 Required Testbed Characteristics

The testbed must have certain properties in order to adequately test the possible hypotheses. First, the testbed must be dynamic and interactive. The scenario must be time varying, thus requiring the subjects to continually update their knowledge base by observing the environment and communicating with the others in the network. The frequent changes also mean the agents must constantly make decisions. The scenario must reflect the consequences of the decision made by the subjects in the network so they have a feedback of the effects of their decisions or indecisions. The testbed must be probabilistic so as to enable uncertainty to be interjected into the scenario. Finally, competition between possible courses of action and interpretations must be arranged, causing the subjects in the network to make tradeoffs or compromises and to choose only a subset of the possible alternatives. The testbed chosen for this report is a simple economic scenario where the decision makers must allocate limited resources to maximize a stated performance index which can then be used to measure the quality of the agents' decision

3.3 Testbed Scenario Generator A simple agrarian economic model was chosen for the limited resources testbed. In this economic system, the decision makers must decide how to allocate land, manpower, money, food, water, and housing in order to optimize the welfare of the economic community. Nature will create random inputs through mechanisms such as weather and land productivity. The model may seem simple on the surface, but once all the relationships have been defined, the reader will see that the model has enough complexity to serve as a functionally adequate testbed. The relationship between problem complexity and system activation is important and will be discussed later. For the sake of illustrative ease, Figure 4 shows only a partial relational flow diagram of the system that would be simulated. The scenario will be time sliced by the four seasons into discrete decisionmaking opportunities for the agents. The agents will make decisions based on the observations made during that season. The scenario generator will take the inputs of the agents, the random inputs from nature, and the current state of the system, combine them and then make available the outcomes and the status of the limited resources to the agents. This information will be the observations for the decisions that are to be made in the next time

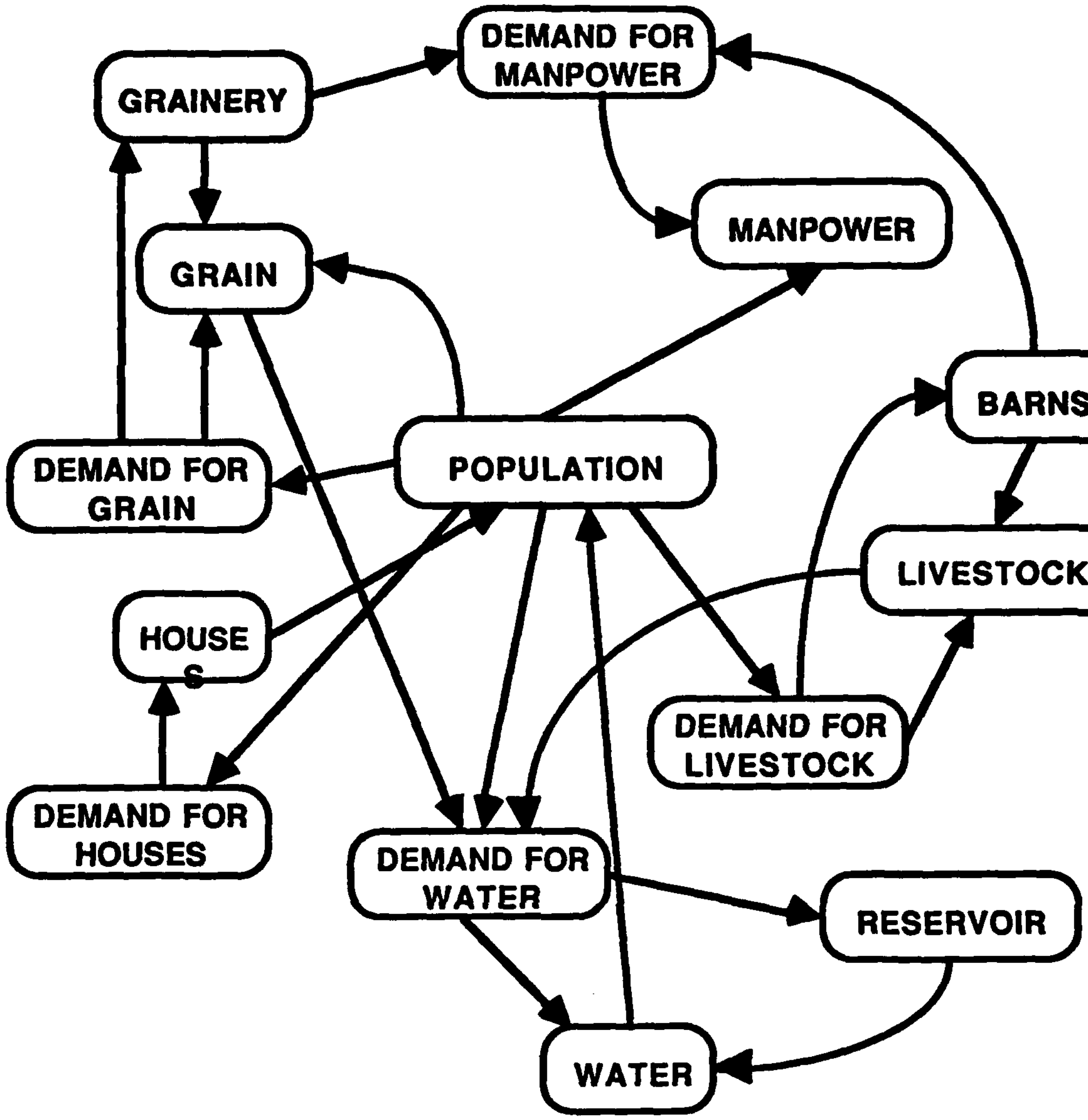
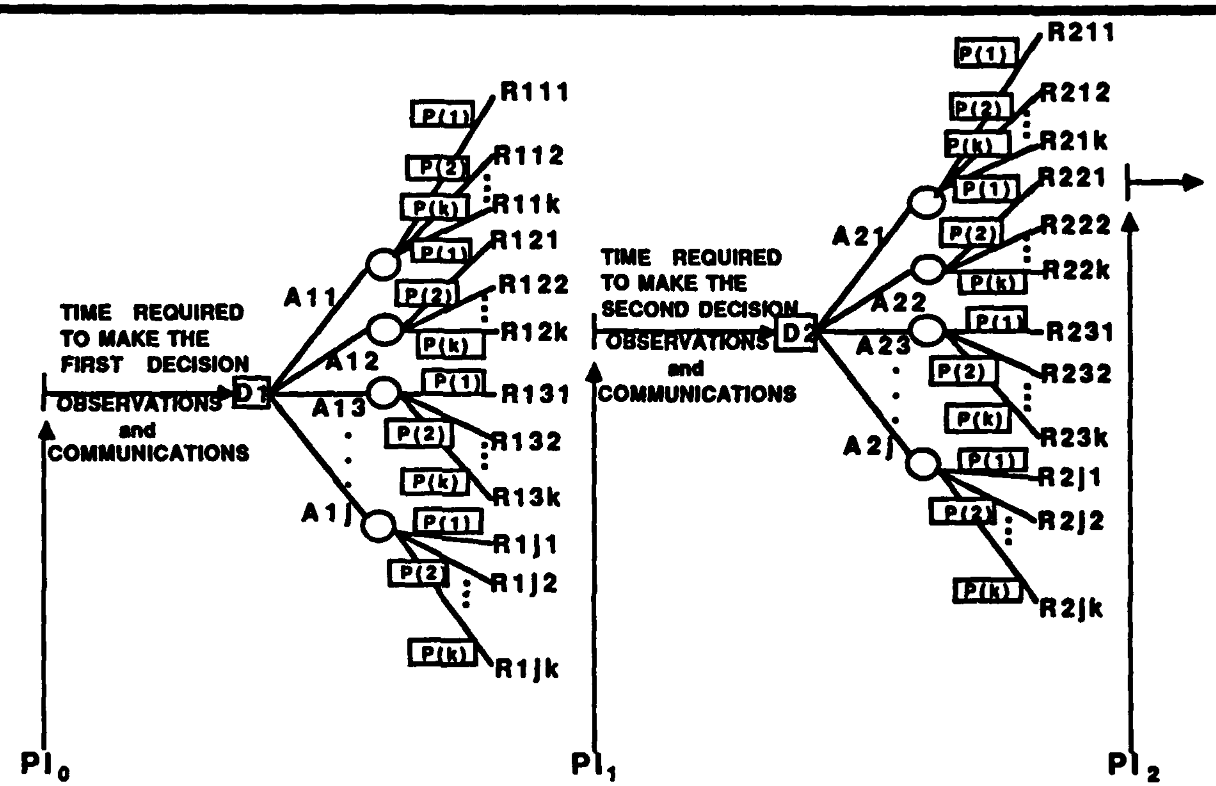


FIGURE 4 - PARTIAL RELATIONAL FLOW OF THE LIMITED RESOURCES TESTBED

slice of the scenario. Figure 5 shows a discrete, functional representation of two decisionmaking opportunities. The testbed is set to some initial conditions which are reflected in the initial performance index PI_0. This and subsequent performance indices are defined as a function of the limited resources available to the economy at any given time. In Figure 5, this collection of resources is represented by R_{ijk}, the resultant bundle of resources from choosing the jth resource allocation alternative at time i given the kth state of nature. Given a bundle of resources, the agents must choose how to allocate group resources in order to maximize group welfare. This decision is represented in Figure 5 as Di, the ith decision. The agents must choose a resource allocation alternative, Aij, while being cognizant of the potential outcomes of the alternatives as a function of the state of nature. The probabilistic character of nature, where p(k) is the probability of the kth state of nature, would define the amount of problem uncertainty. The amount of problem complexity can be changed be varying the level of the problem uncertainty (the probability function of the states of nature) or the number of defined relationships between the scenario parameters. Once the initial conditions are set, the agents must make a resource allocation decision. After each decision a new bundle of resources will be made available to the group. This process would continue until the scenario time limit is reached. The dependence of performance index to the problem solving processing time can be modelled by defining a natural attrition rate for each resource. This would allow the calculation of a continuous, time varying performance index. For now, we can only state that $PI_i = f(R_{ijk})$, but we will be able to expand upon this later.



DI Ith decision

AIj jth resource allocation alternative at the Ith decision making opportunity

p(k) probability of the kth state of nature

RIjk bundle of resources resulting from choosing the jth resource allocation alternative at the Ith decision making opportunity given the occurence of the kth state of nature

PII measure of performance, a function of resources available after Ith decision is made

FIGURE 5 - TESTBED TIMELINE

4.0 DESIGN FEATURES OF THE HYPOTHETICAL TESTBED

4.1 Introduction

The class of decisionmaking structure of interest is agents placed in a goal directed, data driven, cooperative, distributed problem solving network. The issue of interest is the possibility of improving individual and group performance by the introduction of a message categorization system into the distributed problem solving network. The message categorization system will establish feedback loops within the network. These closed loops allow the agents in the network to gain some knowledge of how they are performing, to alter their information processing algorithms or heuristics in order to improve performance, and to learn about each other.

Classical control theory points out the advantages of a closed-loop system over an open-loop one. Though inherently stable, and less costly in terms of components, the accuracy of an open loop system is limited by how well the system input-output relationship can be calibrated. In addition, the open loop system is not suited to handle uncertainty. A *closed-loop* or *feedback control system* does require feedback elements and can become unstable, but the ability of a feedback control system to continually compare the input-output relationship means such a system can be made to perform more accurately than an open-loop system and it has the capability to handle uncertainty. Since reduction of uncertainty is a key design criterion, implementing feedback loops seems to be a natural path to pursue. Figure 6 illustrates the two models of feedback loops the message categorization system will try to establish. If, for example, the feedback loop in the first model was broken, the agents would lose a chance to correct any errors made and the opportunity to learn so as not to repeat the same type of error.

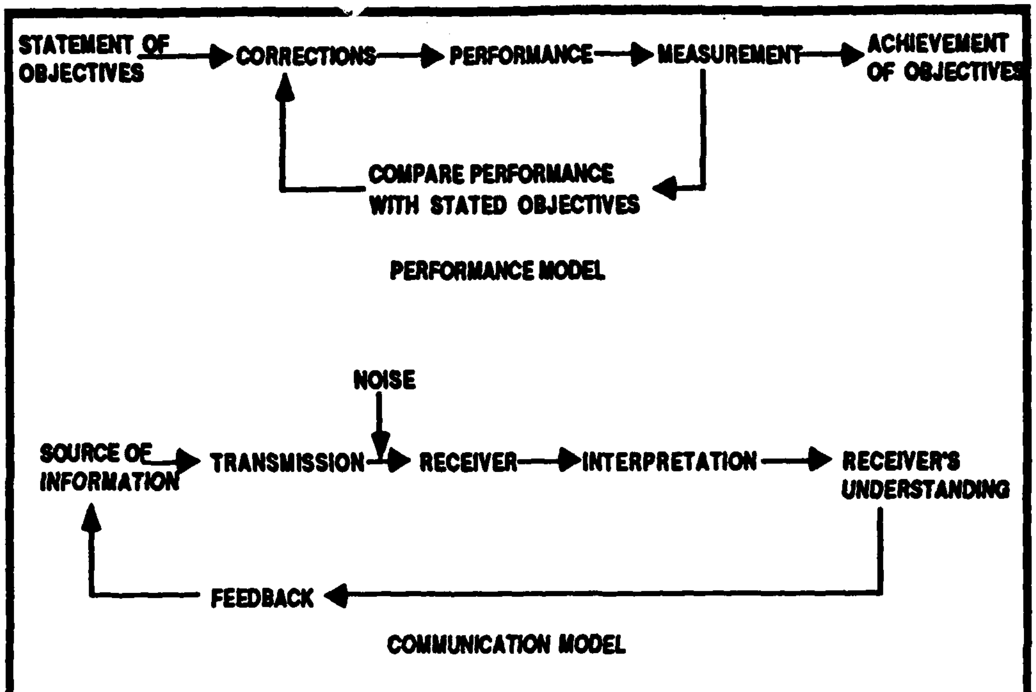


FIGURE 6 - FEEDBACK LOOPS IN THE NETWORK

The message categorization system can be broken down into the following major sub-functions:

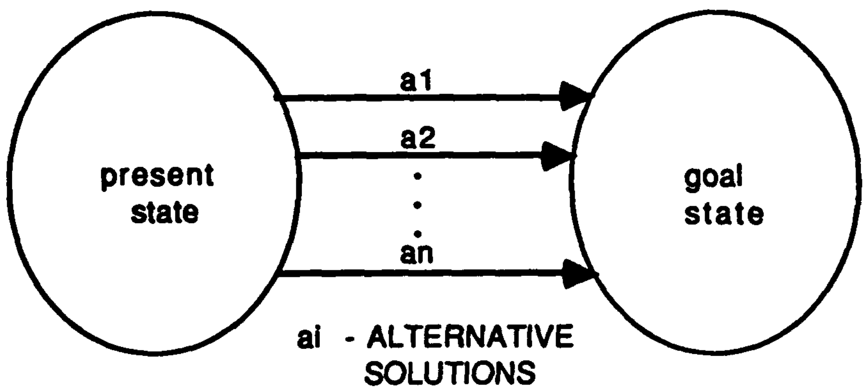
- problem definition and goal setting functions;
- agent-level message prioritization functions;
- system-wide message categorization functions;
- difference resolver functions;
- problem complexity measurement functions;
- *intelligent intermediary* functions.

Each of these functions will be discussed in detail, with examples based on the testbed described above, *in order to clarify or emphasize* points made during this discourse.

4.2 PROBLEM DEFINITION AND GOAL SETTING

4.2.1 Introduction

A simple model of the problem solving process is shown in Figure 7. Problem definition is a key step in the kind of problem solving process being considered here. Definition of the present state and the desired goal state will influence the alternative solutions generated to solve the problem and the criteria for choosing from among the alternative solutions. One must also recognize problem definition is an iterative processand that the problem definition can be changed as the agents gain



1) DEFINE PRESENT STATE
2) DEFINE DESIRED GOAL STATE

THE UNDERSTANDING OF THE DIFFERANCE BETWEEN THESE TWO STATES IS THE PROBLEM DEFINITION

3) GENERATE ALTERNATIVE SOLUTIONS
4) WEIGH ALTERNATIVES AND THEN CHOOSE AND IMPLEMENT THE "BEST" SOLUTION
5) ITERATE PROCESS UNTIL DESIRED RESULT IS OBTAINED

FIGURE 7 - SIMPLE REPRESENTATION OF THE PROBLEM SOLVING PROCESS

experience or as the situation the agents find themselves in changes. Problem definition is more important in a distributed network where one of the basic assumptions made in some cases is meaningful problem decomposition. A partitioning of the problem environment into subsets that lie within an agent's area of expertise

may help offset the costs, and to take advantage of a distributed problem solving network. Generally, before a problem can be decomposed it must be well defined. In such cases, the important outputs of the problem definition phase are:

- group goals
- a weighing function of the goals' relative importance
- a group value assessment function
- a group performance index
- coordination, control, and communication mechanisms

4.2.2 Goal Setting

A key output of the problem definition phase is the set of goals the agents will try to achieve. The goals directly determine how the problem solving process will progress, in both the agent's efforts and in the feedback of the agent's performance. Improper goal setting can be disastrous. Many stories of companies failing due to dogged pursuit of short term gain and profits instead of long term survivability and growth can be found in the news journals. The Russian economy suffered many losses due to expressing production goals of such items as nails and furniture in tonnage instead of more specific quality and quantity measures. Managers pressed to meet the production quotas turned to producing spikes instead of usable nails and inordinately large and heavy furniture started to appear. It is easy to see from these examples why problem solving goals are important. It will be easy to show why problem solving goals will be important in analyzing communications in a distributed problem solving network. We will now look at an example of goal setting from the field of management science as a metaphor to be used in the investigation of distributed problem solving. A management technique that enjoyed popularity for some time was management by objectives (MBO) as advocated by Peter F. Drucker [see Hampton (1980) chapter 8 for

a more in depth coverage of the MBO technique]. In this system the superior would provide the subordinate a framework in which he was to perform. Given this framework, the subordinate would provide a set of goals he would try to accomplish. The superior and the subordinate would negotiate and agree on a final set of goals for the subordinate to achieve. As time passes, the two would get together and review the progress made by the subordinate and make adjustments where necessary. The expected gains of the MBO technique are:

- clear and mutually agreeable goals;
- improved planning;
- clear standard for control;
- improved motivation;
- more objective appraisals;
- better morale.

The possible disadvantages of the the MBO technique are:

- coercion of subordinates;
- approval of incompatible goals;
- additional overhead;
- focus on easily measured results instead of important results;
- rigid pursuit of goals.

We can now apply the lessons of the MBO technique to the distributed problem solving situation. Setting clear and mutually agreeable goals is the primary way the problem will be decomposed. In this proposed testbed, a goal setting phase would be required as part of the problem definition process before the problem solving process will be allowed to start. The agents in the system, as cooperating equals, will assess their current understanding of the problem, assess their current position in regard to the problem, and then define a primary goal and a set of major sub-goals which, if fulfilled, will lead to a satisfactory resolution of the problem. The product of the goal definition phase would be a hierarchical set of goals such as shown in Figure 8. The set of goals can be arrived at by using a simple but effective tool, the relevance tree.

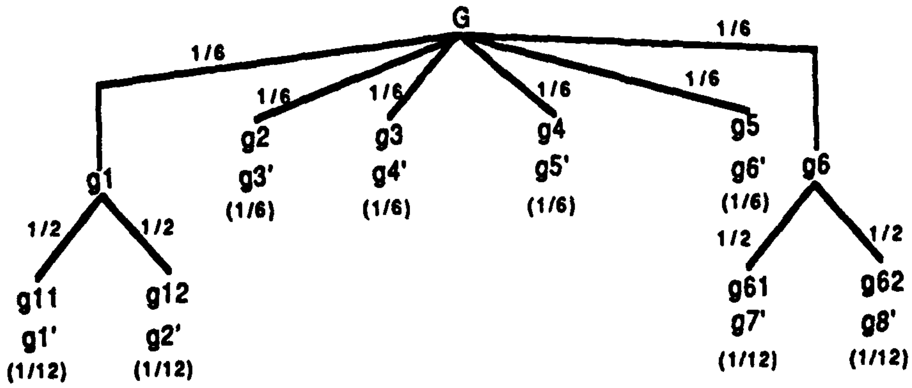
The relevance tree is a tool used to make a choice between competing multi-attribute alternatives, but its structure can be readily adapted for our purposes simply by considering the goals as attributes of the problem to be solved. The relevance tree method attacks the hierarchical goal structure one attribute level at a time.[see Rubinstein and Pfeiffer (1984) for a more detailed presentation of the relevance tree technique] This is done by decomposing the global goal level by level until a level is reached where some objective, observable, and measurable unit can be defined. In our case that would be the set of limited resources such as water, food, manpower. The group will then be asked to assign a relative measure of importance or a rank to each goal. An important feature of this hierarchical set of goals is that a subjective goal such as maximize the welfare of the community can begin to be quantitatively assessed..

4.2.3 Importance Weighing

At a given level, the agents must analyze the goals and assign a relative weight of importance to each goal. Assuming all the goals are relevant in the decision making process, it is easier to assess the relative importance of each goal if the agents can identify the goal which, relatively speaking, is the least important goal at that level and assign it a value of one. Using this goal as a reference, pairwise comparisons of the other goals with this reference goal will yield relative weights for the remaining goals at

that level. Consistency of the assigned weighing values can be checked by making pairwise comparisons across all goals at that level and making adjustments where necessary. By dividing the individual weights by the sum of the weights at that level, a fractional value for the relative importance of an individual goal can be found. A goal having the value of six-tenths for instance would indicate that this particular goal was assigned the majority of the measure of importance at this level. A simple example where all goals at any given level are equally important is shown in Figure 8 to illustrate the mechanics of the process. Six goals, g1 through g6, have been defined at the first level. If they are of equal importance, they would each have a weighing value of one sixth. Another level of goals which must be evaluated exists under g1. Since there are only two goals at this level, each goal has a weighing value of one half. This process would be repeated until all levels of goals have been exhausted. After each level of goals has been assessed, the product of the weighing values along each path from the global goal to a measurable resource unit can be calculated and used as a measure of relative importance for each resource. The product of the weighing values will be defined as W_{im}, the importance weighing value of the mth resource at time i.

MAJOR GOAL - INCREASE COMMUNITY'S WELFARE	[G]	
SUB-GOALS -		
INCREASE FOOD SUPPLY	[g1]	
INCREASE GRAIN STORES	[g11]	[g1']
INCREASE LIVESTOCK HERD	[g12]	[g2']
INCREASE HOUSING	[g2]	[g3']
INCREASE MONEY STORE	[g3]	[g4']
INCREASE WATER SUPPLY	[g4]	[g5']
INCREASE POPULATION	[g5]	[g6']
INCREASE PRODUCTION	[g6]	
INCREASE MANPOWER SUPPLY	[g61]	[g7']
INCREASE MANPOWER PRODUCTIVITY	[g62]	[g8']



EACH GOAL MUST BE MEASURABLE:

g1' - x bushels or x bushels/person
g2' - x heads or x heads/person
g3' - x number of single unit dwelling or x people/dwelling
g4' - $
g5' - gallons reserve
g6' - population or population growth rate
g7' - x manhours (mh) available
g8' - manhour productivity index =
f(bushels of grain/mh, land tilled/mh,acres of land cultivated/mh,...)

FIGURE 8 - GOAL SETTING

4.2.4 Value Assessment

The next measure that the agents must define is the value associated with the current level of the measurable unit, a measure of utility if you like. For instance, if the amount of grain available is in excess of the yearly community requirement by a factor of two, the agents might assign the maximum utility value to this resource. If the available supply of grain is dangerously low, the lowest utility might be assigned. An example of utility assignment for each resource is shown in Figure 9. A more detailed and classical treatment of utility theory can be found in Keeney and Raiffa (1976). The table in Figure 9 gives an example of the agents' definition of the relationship between the physical measure of a resource, q_{im}, and the utility value of that resource, U_{im}. The system now has enough information to establish some form of performance index.

GOAL

UTILITY

u(i') \ gi'	g1'	g2'	g3'	g4'	g5'	g6'	g7'	g8'
1	q(1)≤c(1) or q(1)>s(1)	q(2)≤c(2) or q(2)>s(2)	q(3)≤c(3) or q(3)>s(3)	q(4)≤c(4) or q(4)>s(4)	q(5)≤c(5) or q(5)>s(5)	q(6)≤c(6) or q(6)>s(6)	q(7)≤c(7) or q(7)>s(7)	q(8)≤c(8) or q(8)>s(8)
2	q(1)≤2c(1) or q(1)>c(1)	q(2)≤2c(2) or q(2)>c(2)	q(3)≤2c(3) or q(3)>c(3)	q(4)≤2c(4) or q(4)>c(4)	q(5)≤2c(5) or q(5)>c(5)	q(6)≤2c(6) or q(6)>c(6)	q(7)≤2c(7) or q(7)>c(7)	q(8)≤2c(8) or q(8)>c(8)
3	q(1)≤3c(1) or q(1)>2c(1)	q(2)≤3c(2) or q(2)>2c(2)	q(3)≤3c(3) or q(3)>2c(3)	q(4)≤3c(4) or q(4)>2c(4)	q(5)≤3c(5) or q(5)>2c(5)	q(6)≤3c(6) or q(6)>2c(6)	q(7)≤3c(7) or q(7)>2c(7)	q(8)≤3c(8) or q(8)>2c(8)
4	q(1)≤4c(1) or q(1)>3c(1)	q(2)≤4c(2) or q(2)>3c(2)	q(3)≤4c(3) or q(3)>3c(3)	q(4)4c(4) or q(4)>3c(4)	q(5)≤4c(5) or q(5)>3c(5)	q(6)≥4c(6) or q(6)>3c(6)	q(7)≤4c(7) or q(7)>3c(7)	q(8)≤4c(8) or q(8)>3c(8)
5	q(1)≤5c(1) or q(1)>4c(1)	q(2)≤5c(2) or q(2)>4c(2)	q(3)≤5c(3) or q(3)>4c(3)	q(4)≤5c(4) or q(4)>4c(4)	q(5)≤5c(5) or q(5)>4c(5)	q(6)≤5c(6) or q(6)>4c(6)	q(7)≤5c(7) or q(7)>4c(7)	q(8)≤5c(8) or q(8)>4c(8)
6	q(1)≤6c(1) or q(1)>5c(1)	q(2)<6c(2) or q(2)>5c(2)	q(3)≤6c(3) or q(3)>5c(3)	q(4)≤6c(4) or q(4)>5c(4)	q(5)≤6c(5) or q(5)>5c(5)	q(6)≤6c(6) or q(6)>5c(6)	q(7)≤6c(7) or q(7)>5c(7)	q(8)≤6c(8) or q(8)>5c(8)
7	q(1)≤s(1) or q(1)>6c(1)	q(2)≤s(2) or q(2)>6c(2)	q(3)≤s(3) or q(3)>6c(3)	q(4)≤s(4) or q(4)>6c(4)	q(5)≤s(5) or q(5)>6c(5)	q(6)≤s(6) or q(6)>6c(6)	q(7)≤s(7) or q(7)>6c(7)	q(8)≤s(8) or q(8)>6c(8)

q(m) = physical measure of the mth resource
c(m) = lower critical threshold level of the mth resource
s(m) = upper saturation level of the mth resource

FIGURE 9 - SAMPLE OUTPUT OF THE UTILITY ASSESSMENT PROCESS

4.2.5 Performance Index

A measure of group performance can only be established by combining the utility measure of a resource and the importance weighing assigned to the resource. R_{ijk} is represented by the accumulation of q_{im}. However each resource has a different unit of measure so it is difficult to directly work with R_{ijk}. The agents' defined utility function will be used to transform the various physical units of measure to a common utility unit of measure. Finally, the performance index can be found by taking a weighed summation of the resource utilities, namely $PI_i = SW_{im}U_{im}$. The performance index can be updated any time a physical measurement is taken, the importance weighing changes, or the utility definition changes. The desire to explicitly indicate the time varying property of the utility value and the importance weighing value for each resource is the reason both values are indexed by time.

4.2.6 Summary of the Problem Definition Phase

Once the goals and their relationships have been defined, the agents will be reminded that the goals, their importance weighing, or the value assessment are not a rigid set and can be changed to meet changing circumstances. Any agent in the network can call for goal reassessment. The system may also intervene as an intelligent intermediary and suggest a re-evaluation of the goals. In the hypothetical testbed, this might happen if the supply of a critical resource , such as water, became low. This would be accomplished by having the agents define a critical threshold level, c(m), such that if $q(m) < c(m)$, then the system will give an automatic warning. In our example, the system would flag the condition and suggest raising the importance assigned to the water supply goal. The system might also intervene if some resource was in over supply or at a saturation level, $q(m) > s(m)$. An example of this case would be the grain supply becoming so large that storage space was exhausted. The system might suggest lowering the relative importance of this goal and concentrate on another

area that is not performing so well. Having too little or too much of a resource might diminish the welfare of the community.

The output of the goal setting process is:

- main objective;
- sub-goals supporting the main objective;
- goal relationships (priority);
- measures for goal assessment.

These outputs will provide:

- mechanism for problem decomposition and task coordination;
- basis for information routing in the network;
- standards for measurement and control (performance index);
- direction for data interpretation.

The above discussion summarizes the process behind goal definition, importance weighing generation, and utility measurement definition. For practical experimentation purposes, the three factors just mentioned would probably be best defined by the experimenter. By holding his set constant, the possible intervening effects of these variables can be controlled. A more important result of predefining these variables is that we avoid the Arrow Paradox in the beginning stages of the experiment. This well known condition points out the possible violation of the law of transitivity in a group decisionmaking environment when no control hierarchy other than majority rule is defined [see Rubinstein (1975) for a more detailed account of this phenomenon]. Although this paradox is an important phenomenon, the testbed is not suited to handle this condition and as such we will try to avoid it by predefining the goals, their relative values, and utility measure for each resource. A rationale

explaining the choice of the final set of goals, importance weighings, and utilities will also be supplied This reasoning should also serve as a beginning to a set of guidelines to help in task execution. The problem definition phase of the testbed experiment would be used to explain the concepts behind and the functions of the three parameters and to assure that the subjects understand their importance.

4.2.7 Potential Effects of Goals on Coordination, Control, and Communication in the Network

Now that the problem, group goals, and a method to gauge group performance have been defined, the problem of coordinating task distribution in the network can be addressed. A simple decomposition rule could be to assign each agent a sub-goal to execute based on his area of expertise or even on a volunteer basis. Each agent knows what his job is and what the other agents are working on. Each agent arrives at a solution for his particular area. These sub-solutions are then merged to give an overall solution to the problem. In addition to decomposing the problem, the goals also provide a mechanism to coordinate the information routing in the network. Each agent knows what information the others may be looking for based on the goals they have been assigned. As the agent comes upon information, he can route it to those he feels will find it of some use.

A more complex problem decomposition would be to assign each agent a goal to fulfill and to allow him to assign sub-tasks to other agents. This is the concept Smith and Davis try to establish with their contract net mechanism described above. The first agent still has the responsibility of fulfilling the goal and he may indeed still be performing a majority of the sub-tasks necessary to meet his objectives, but now he is acting as the coordination center of the tasks centered around his area of responsibility. Communication is not as easily achieved since an extra layer or layers can been placed between where useful information exists and where it is needed. The same is true for control due to the possibility of a separation between where the decisions are being made and where the performance is being measured. Coordination is similarly

hampered since the center of responsibility can be removed from the point of task execution. However, communication, control, and coordination are possible since a traceable path has been established. The problem is how to streamline the coordination process so that a responsible agent spends less time managing his tasks and more time at actually executing his tasks.

Control, in the cooperative sense, is achieved in both examples above by assigning definite and known areas of responsibility. Individual agent performance indices can be defined as a subset of the overall performance index based on the areas of responsibility assigned to particular agents. If an individual agent has contracted out sub-tasks, a further partitioning of the performance index can be used to measure the performance of the agent who has taken on the sub-task. In any case, the performance index can be used to evaluate the agent's performance and to provide feedback to the agents so they can better control the output of their efforts or that of any contractor under them. This is a way to affix accountability to an agent and induce him to act in accordance to the groups wishes.

Now that accountability has been established, the agents have to be motivated. Beyond peer pressure, incentive can be established by tying a reward system to the global performance index. If the reward is distributed as a function of the sub-goals, each agent can be induced to perform to the best of his abilities. Each agent has an incentive to act in a cooperative manner in order to increase the global reward level. Each agent has incentive to take on the tasks he is most suited for and to take on as many of theses tasks as possible. Adelman et al (1986) have presented preliminary experimental results which show motivation or incentive was a significant independent variable in improving group performance.

The outputs of the goal setting phase would appear, logically,to be the first step in decreasing the coordination and control problems within a distributed network if one adopts the sequential problem solving model advocated here. Problem decomposition can be extended by defining additional layers of sub-goals, but the important point here

is goals can be used to decompose problems, to assign areas of responsibility to specific agents, and to establish means of coordinating and controlling efforts and information flow in a distributed network. The system can be empowered to further facilitate coordination and control. The system can monitor the goal setting process and know what the group goals are, the relative importance of each goal, and which agents are responsible for which goals. If the system could somehow discern the information content of the messages as they are transmitted through the network and the information preferences of the agents, it could start functioning as an intelligent intermediary. The system would have enough knowledge about the agents' goals, responsibilities, and preferences to effect more efficient communications in the network. The obvious next step is to have the agents classify certain message features and to list their information preferences.

4.3 MESSAGE CATEGORIZATION

4.3.1 Introduction

At the heart of the proposed system is an on-line message categorization mechanism. The categorization mechanism has two major units. The first unit performs an information preference definition function. The second unit performs a message assessment and categorization function. Each of these units will be discussed individually along with a discussion on how the system can use the information from each of these units to intelligently interact with the agents in order to improve network performance.

4.3.2 Defining Agents Preferences

The information preference function requires the agents to state what types of information they believe that they need to execute their tasks. As the agents gain experience they should become better at accurately defining their information requirements. If the agents rank their preferences, then incoming messages can be prioritized and queued based on a weighted average of identified message features such as relevant goals, message priority, sending agent and elapsed time. In the beginning when uncertainty in the system is high and confidence among the agents is low, much of the agent's time will be spent in a discovery process. The agent is unfamiliar with the other agents in the system, the types of tasks that need to be done, the types of information available, the reliability of the system, et cetera. Each incoming message would have to be given equal weight until a discrimination process can be established. The agent has to develop a systematic method of screening incoming information in order to reduce wasted time and effort.

The best way to do this is to become familiar with the other agents in the system and evaluate their abilities as information sources and then use the group defined goal priorities to rank task importance. For example, suppose agent A is in charge of the water supply and two messages dealing with the water goal having the same priority

arrive, one from agent B and the other from agent C. At the start of the problem solving process agents B and C are perceived by agent A as equally unknown entities; therefore discrimination between the two is not possible. As the agents interact, agent A can develop models of agents B and C. For instance A may have learned that C has a tendency to overestimate the importance of his information and this is reflected in the unduly high priority he assigns to his messages to the other agents. B, on the other hand, has been found to be consistently accurate in his priority assignments to his outgoing messages. All other things being equal, A should prefer to see the message from B. Now suppose A receives two message from B, the first dealing with the food goal and the second with the water goal, but they both have the same priority rating. In order to discriminate between these messages, A must have previously analyzed his tasks and defined which goals he is fulfilling with his efforts. If for example increasing food production is his major responsibility while the water supply is another agent's concern, he should prefer to see the first message over the second. Taking the example one step further, suppose A is equally responsible for both the food and water goal. Discrimination between the two messages from B can only be done if A ranks the goals. A already has the group-defined priorities from the goal definition phase discussed above at his disposal as a discriminator. All A has to do is periodically review the ranking and assure that it is still appropriate. A can also use the utility measure of a goal to determine whether the resources are at an acceptable level or if an emergency situation exists (e.g. no water available). Taking the example to the final step, suppose A received two messages from the same agent with the same priority, dealing with the same goals. The only way to discriminate between the two messages is the staleness of each message. A may define a preference for newer messages since they contain the latest information.

The situations described above are probably the simplest examples of what could happen within the network. For the network to work at such a simple level, A must understand the goals and how his tasks relate to them. A must also have a fairly

accurate knowledge of the systems current state. A must learn not only how discern between the other agents in the network, A must learn to identify if an individual agent treats separate goals, tasks, or types of information differently from each other. The simple model quickly becomes complex. An agent must learn about himself and the others in the network so he can define his preferences and begin the process of filtering incoming information.

Establishing the user defined, dynamic, prioritizing buffer to order incoming messages in accordance to the agents' preferences or needs is an important development in the lifecycle of a distributed problem solving network. Mechanisms that will accelerate the process are undoubtedly important The system also uses this feature to continuously gather data on the agent's current areas of interest, the goals he is actually working towards, the type of data he needs, the agents he wishes or needs to interact with, et cetera. With this information, the system will be able to intelligently interact by helping to route information to agents who may have been overlooked or help lessen the load on the communication lines by waiting for idle time to send messages containing subject material to which the receiving agent has already assigned a low priority. The functional representation of how the agent's workspace might be partitioned is shown in Figure 10. The system now knows what type of information the agents would like. Next, it must determine the information content of the messages in order to make use of this information.

4.3.3 Message Evaluation

The possible message features must be identified and then the task of laying out the message categorization design features system can begin. Message features are a basic component of the categorization system and as such, they must be defined. Message features that have been identified are:

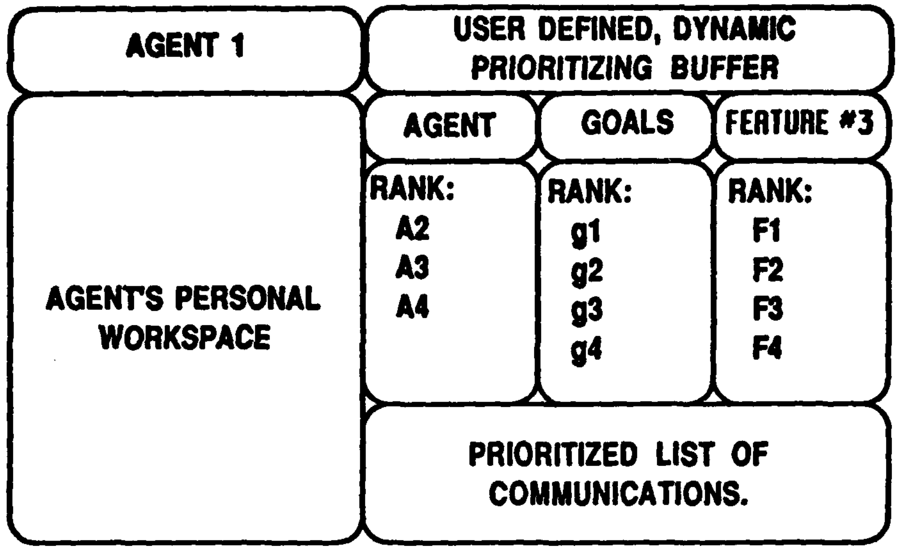


FIGURE 10 - FUNCTIONAL DEPICTION OF AN AGENT'S WORKSPACE

- agents sending the information;
- assigned message priority;
- assigned message goals;
- message type;
- message staleness.

Of the four features listed, the agents must evaluate the message priority, message goals, and message type.

The message assessment and categorization unit requires agents, both sending and receiving, to independently assess each message for information content and then categorize the message using the features mentioned above. Acting as a central clearinghouse for all messages sent on the network, the system will collect unbiased agent assessment of the information available to the network. These data will be used to generate a new piece of information, the degree of disagreement between agents on the particular features of the message information. The hypothesis of this report is that this new piece of information is very valuable in establishing control and coordination

in a distributed network through numerous dynamically formed feedback loops. The degree of disagreement can be used to drive additional dialog between the agents until the disagreement is resolved. An immediate consequence of this difference resolution, if it occurs, is a better assessment of the information. A long term benefit of this process is that by continually comparing final assessments to original assessments, each agent can monitor his own performance and the performance of the other agents in the system. This, when it works, is the primary way for each agent to learn about himself and the other agents in the network and to reduce one controllable source of uncertainty in the network. The desired outputs are discrimination rules based on the message sending agent, streamlined communications, and more time for agents to execute their tasks.

Although the agents may have agreed to the overall worldview (the group goals), they will not necessarily have the same interpretation of a piece of information; thus the existence of the difference resolution process. This difference resolution process does not necessarily converge. However, the process must terminate if an impasse is reached in order to free the agents for other tasks and to eliminate unnecessary communication. There must be a procedure that allows the agents to agree that they disagree. The final opinion of all agents involved must be recorded and made available to the rest of the network. Even without convergence, the agents have an opportunity to learn about each other.

The system can also monitor each agent in the system by maintaining a running statistical profile of each agent and pinpoint potential problem areas. Examples of problems would be agents who tend to cry wolf, agents who hoard information, or agents who send messages that unnecessarily tie up communication resources. The system is dynamic as it can adapt to changing agent preferences as quickly as the agent can enter the information into his data base. The system can monitor an agent's stated preference and the actual information feature the agent seems to be favoring. The system can also monitor an agent to see if he is searching for information that is

consistent with his areas of responsibility or if he may be able to take on additional responsibility. These examples are just two ways the system can intervene intelligently and enhance the communication, control and coordination in the network. We turn now to a discussion of the message features starting with message types.

Message types that have been identified so far are:

- dialog messages;
- system messages.
- messages requesting help;
- messages requesting information;
- response to requests;
- messages relaying information from observations;
- messages relaying recommendations/decisions;
- messages relaying results/conclusions;

The main reason to distinguish message types is this information can be used as a discriminator. Imagine a situation where an agent is expecting an important result or piece of information from another agent. The first agent can raise his preferences for these types of messages from a particular agent in order to identify the particular messages more quickly.

Another reason to identify message types is not all messages need be evaluated for their information content. For example, two agents who have been brought together by the system to resolve differences should not have to evaluate the dialog that is necessary to converge on a mutually agreeable interpretation. System messages will also be exempt from evaluation since they are used only to issue warnings, to ask for clarification, or to request or give network status.

The final important reason message types must be identified is that the time history of the message types sent will be useful in analyzing the network. Figure 11 shows a matrix representation of data that could be collected. For example, if the message

categorization system were effective, one would expect the frequency of entries in the left-hand portion of the matrix to decrease as a function of time. A plausible explanation for this observation would be the agents are learning about each other and that they are sending information to those agents who require it without being asked to do so. There may be other process that such data could reveal.

ith DECISION MAKING OPPORTUNITY	REQUEST FOR HELP	REQUEST FOR INFORMATION	DIALOG MESSAGES	SYSTEM MESSAGES	RESPONSE TO REQUESTS	RELAYING INFORMATION	RELAYING RESULTS OR CONCLUSIONS	RELAYING RECOMMENDATIONS OR DECISIONS
AGENT 1								
AGENT 2								
AGENT 3								

FIGURE 11 - DATA OF MESSAGE TYPES SENT BY AGENTS DURING ONE DECISION MAKING TIME PERIOD

The next feature for which a message is evaluated is the importance of the information or the priority of the message. The categorization mechanism requires the sending agent to assign a priority level to outgoing messages. An example set of priority classification would be subjective descriptors such as 1 - critical, 2 - high, 3 - important, 4 - useful, 5 - routine, or 6 - low. The assigned priority can be used by the system to route messages. All other factors being equal, the higher the message priority, the sooner the message will be transmitted on the network. However, all other things may not be (and are probably not) equal. The priority of a message in the network could be adjusted by the known history of the sending agent, the current state

of the resource in question, or by the indicated preferences of the receiving agents. This means a priority rating of 3 does not always carry the same weight. This is true across all the agents at any given time or particular to a single agent from one time to another. The meaning of a priority rating can be adjusted by the current state of the network.

The last message feature an agent must evaluate is the goal to which the information pertains. In our example, this would be choosing from the set of eight end goals shown earlier in Figure 8. Complexity can be added to the goal evaluation process by allowing multiple goal assignment or even a ranked multiple goal assignment.

As stated before the above information is stored in a central database by the system. As other agents get the information, they are required to assign a priority and goal assessment to the message. The system stores this information in the database along with the original piece of information. The start of the chain of events for the transmission of a single message is shown in Figure 12. This figure is a functional representation of the window an agent would see when he sends a message. By properly responding to the system prompts, the message will be allowed on the network and the sending agent establishes a record in the database with the following information:

- sending agent;
- receiving agent or agents;
- time message was sent;
- the message as sent;
- the sending agents message evaluation.

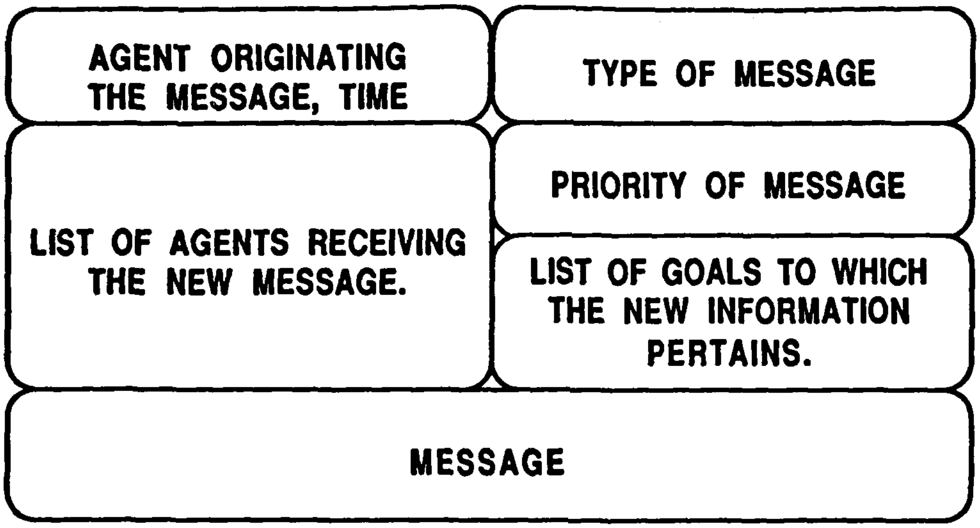


FIGURE 12 - FUNCTIONAL REPRESENTATION OF THE MESSAGE SENDING WINDOW

After a receiving agent reads and evaluates the message, another record is established in the relational database corresponding to the above database entry. This record would contain the following information:

- receiving agent;
- sending agent;
- time message was read;
- the message as received;
- the receiving agents message evaluation.

Figure 13a shows the events that would lead to the formation of the above two database records. The process is started when a message, M11 [Mij = ith message from jth agent], is sent. In this example, the sending agent is required to assign a message priority rating, P111 [Pijk = priority assigned by the kth agent to the ith message from the jth agent].The sending agent is also required to state which goal or goals the information in the message impacts and enter a goal assessment, G111 [Gijk = goal assessment assigned by kth agent to ith message from jth agent].The information from the sending agent is passed through the system manager and stored in a central database and then the message and only the message is sent to the agents in the network that were selected to receive the message, see Figure 13b. An agent can be selected by the sending agent or the system. At this point the message is now entered on the message queue in the individual agent's personal workspace. For the sake of this example suppose two agents in the system were chosen to receive the message and that they both choose to read this message, the system will require each receiving agent to assign to the message a priority rating (P112 from second agent and P113 from the third agent) and a goal impact assessment (G112 from the second agent and G113 from the third agent) . The system will store the information from the receiving agents along with the time the agents read the message in the central database, see Figure 13c. The system calculates the differences between the agents' assessments (DP_{ij} and DG_{ij}) and stores this information, see Figure 13d. Skipping for now the discussion on difference resolving and assuming convergence, Figure 13e shows the final system state with respect to message M111 after all the agents have come to agreement on how to classify the information in the message. P_{ijF} and G_{ijF} are the final group priority and goal evaluation of the ith message from the jth agent.

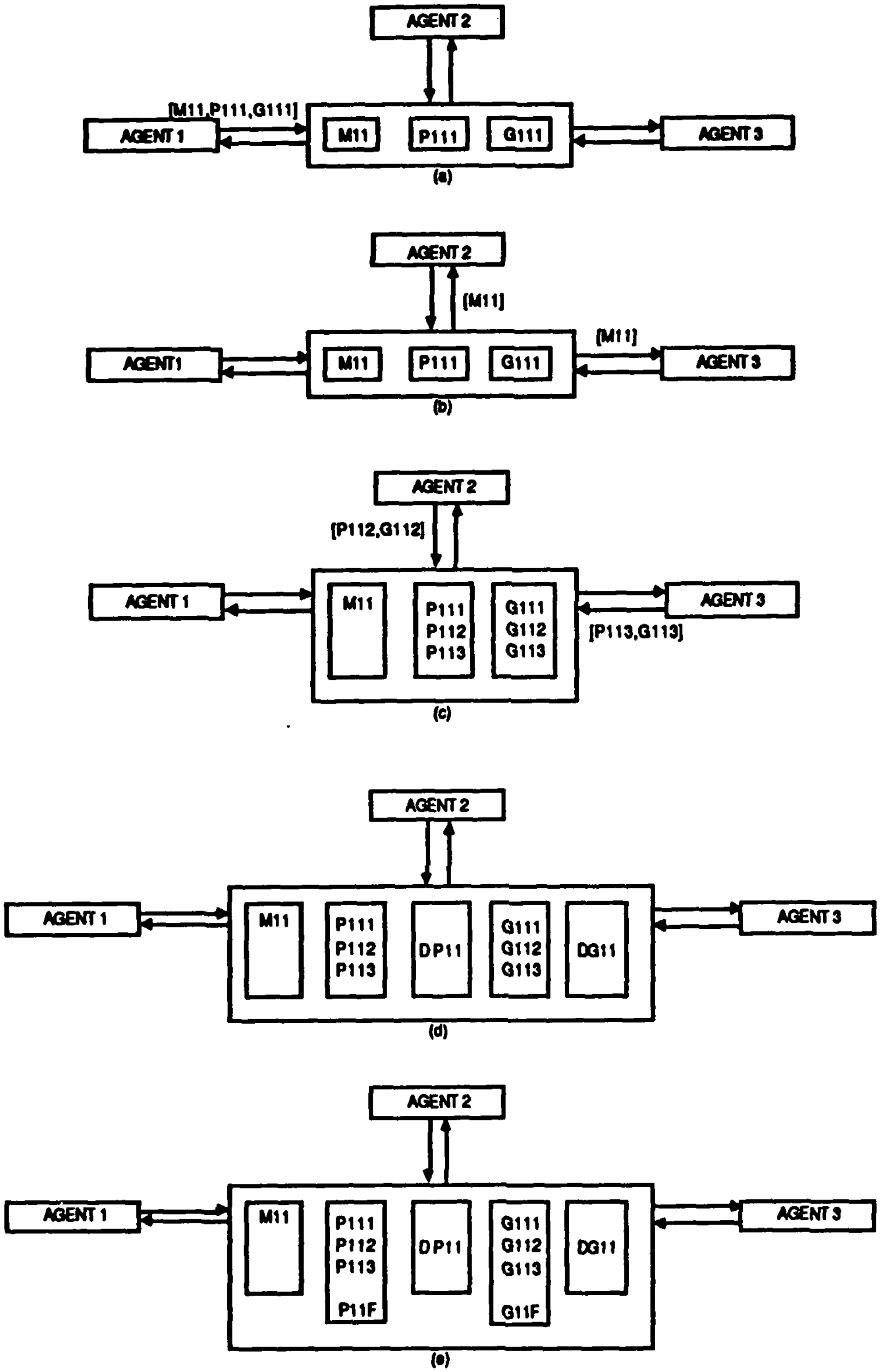


FIGURE 13 - MESSAGE SENDING IN THE MULTIPLE AGENT NETWORK

4.4 DIFFERENCE RESOLUTION

The next question that has to be answered is how to match the agents in order to resolve differences of opinions. The information must somehow be used to help agents initiate constructive dialog to work out differences and arrive at a better assessment of the information. Many parameters can influence how this may be done and it is not clear which method is preferable. This is a complex issue since a multiple number of agents may be involved and the matter of timing and precedence begins to complicate the process. In order to illustrate the complexity of the resolving differences, a simple three agent network example will be given.

$$\begin{array}{cc} & \begin{array}{ccc} \mathrm{M1} & \mathrm{M2} & \mathrm{M3} \end{array} \\ \begin{array}{c} \mathrm{M1} \\ \mathrm{M2} \\ \mathrm{M3} \end{array} & \begin{bmatrix} \mathrm{DM11} & \mathrm{DM12} & \mathrm{DM13} \\ \mathrm{DM21} & \mathrm{DM22} & \mathrm{DM23} \\ \mathrm{DM31} & \mathrm{DM32} & \mathrm{DM33} \end{bmatrix} \end{array}$$

M1 - MESSAGE EVALUATION OF AGENT 1
M2 - MESSAGE EVALUATION OF AGENT 2
M3 - MESSAGE EVALUATION OF AGENT 3

DMij = Mi - Mj - DIFFERENCE BETWEEN THE MESSAGE EVALUATIONS OF AGENT i AND AGENT j

DMij = 0 FOR $i = j$;
DMij = Mji FOR $i \neq j$

FIGURE 14 - AGENTS MESSAGE EVALUATION DATA

Suppose Agent 1 sends the same message to Agent 2 and to Agent 3. If 2 and 3 simultaneously respond to the message we will have the data shown in Figure 14. The rows and columns headers (M1, M2, and M3) represents each agent's evaluation of the message. The entries in the matrix represents the difference between the agents' assessment of the message. How should the difference be resolved? Some possible difference resolving rules are:

- provide the agents with a group statistic of the assessments;
- inform the agents of the summary of all assessments;
- introduce agents with divergent views to ones with views closer to group norm;
- introduce the agents with the more opposing divergent views to each other.

To add complexity to the previous situation suppose 2 responded before 3 and that 1 and 2 came to an agreement on a new message evaluation and then 3 responded. The situation is depicted in Figure 15. The question to be answered is how to get to from state 3 to state 4. Would a different result occur if a dialog were established between 2 and 3 instead of between 1 and 3? Do we risk "groupthink" and stifle creativity by allowing 3 to know that he is the only dissenting agent? Do we use the notion of cognitive dissonance and establish a dialog between 3 and the agent that has changed his opinion the most or is this a further potential for groupthink. As stated above a clear choice is not obvious. However a potentially wealthy source of data to study this area seems to exist. Figure 16 shows the type of data that could be collected. A major function of the testbed would be measuring group performance as the difference resolving rule is systematically varied and to collect the data outlined above in order to analyze the differences.

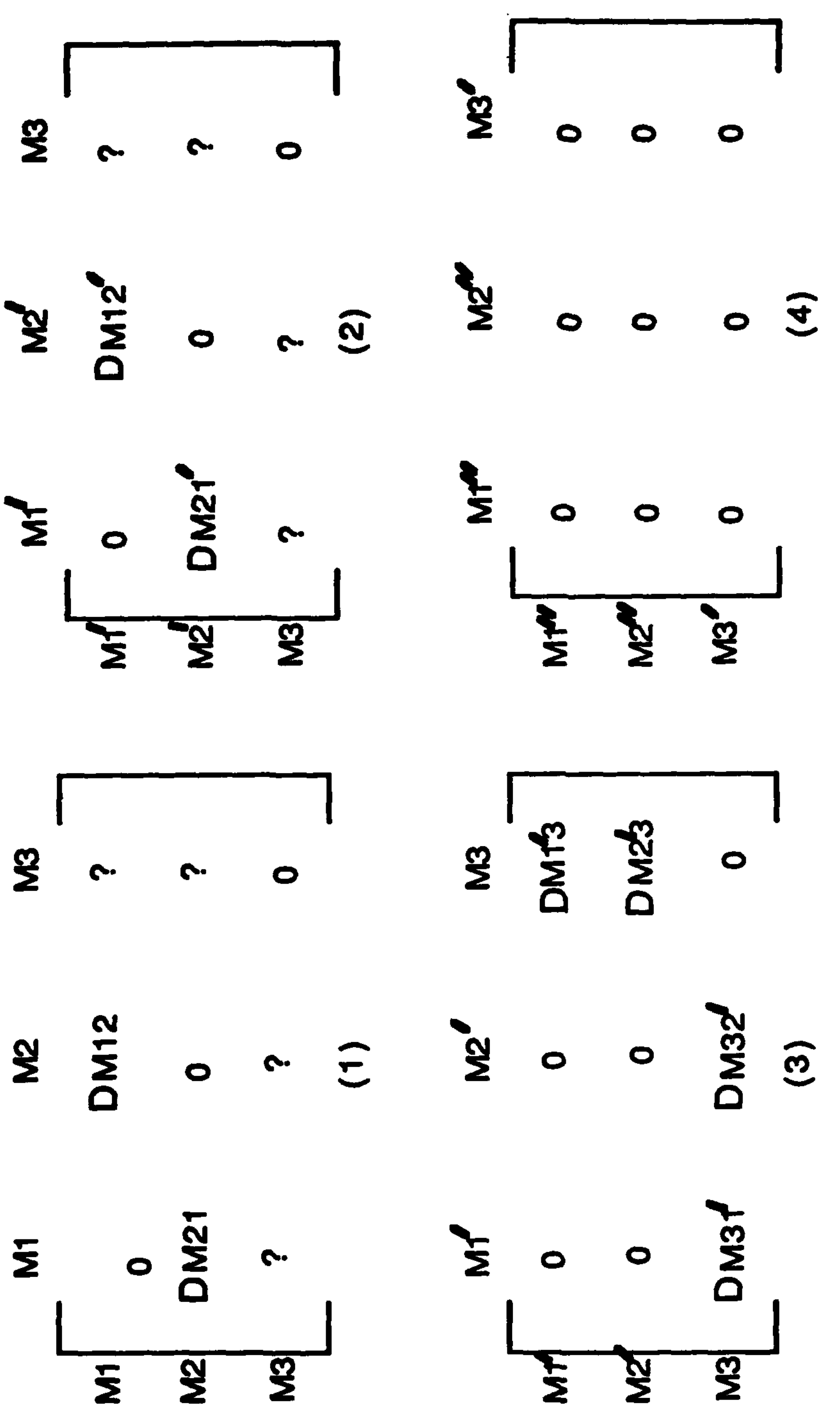


FIGURE 15 - MULTIPLE AGENT DIFFERENCE RESOLVING WITH TIME SEQUENCING EFFECTS

4.5 SYSTEM ACTIVATION

The question of when to implement the type of system that is being proposed was mentioned earlier. Certain classes of problems are too simple or routine to require

such an elaborate system. Problems characterized as being repetitive, under static or slowly varying conditions, with low uncertainty, and adequate

MESSAGE EVALUATION	MEASURE OF DIFFERENCES OF MESSAGE EVALUATION
E1 E2 E3	DE12 = DE21
E1¢= E2¢	DE21¢=DE12¢= 0
E11≤ = E22≤ = E33≤	DE11¢
	DE22¢
	DE31¢=DE1¢ß = DE32¢= DE2¢ß
	DE13 = DE31
	DE23 = DE32
	DE33¢
	DE2≤3¢= DE1≤3¢= DE3¢2≤ = DE3¢1≤
	DE1¢1≤ = DE1≤1¢= DE2¢2≤ = DE2≤2¢
	DE11≤
	DE22≤
	DE31¢=DE1¢ß = DE32¢= DE2¢ß
	DE13¢=DE3¢1
	DE23¢=DE32¢

FIGURE 16 - COLLECTABLE MESSAGE EVALUATION AND DIFFERENCE INFORMATION

data are sometimes classified as programmed decisions. Little risk is associated with these problems since past experience has made solving such problems a mechanical exercise of inputting the data in to a well proven algorithm and cranking the handle until a solution achieved. The overhead costs of the proposed system would outweigh the potential performance increase promised by such a system for this type of problem. A certain level of complexity must exist before the net effect of implementing the proposed system would return a positive yield. For a self starting system, a threshold level of complexity, t(c), must be defined and a measure of complexity, m(c), greater than the threshold level must be sensed by the system before the system transitions

from a role as a monitor to an active intelligent intermediary. The problem complexity can be measured by monitoring such things as:

- the number of states of nature and the associated probability functions;
- number of limited resources and competing uses;
- number of agents in the system;
- number of goals.

The system can gain most of this information by monitoring the problem definition stage of the process. This is another reason the problem definition stage is required and important. An interesting experiment for the testbed would be to measure group performance while systematically varying the problem complexity. Intuitively, one expects the results to be of the form of the classic U-shaped curve of Figure 17. Verifying this assumption is important since the knowledge of this relationship is necessary if the proposed system is to ever evolve beyond a testbed vehicle. One has to know when the problem warrants such an elaborate system.

4.6 INTELLIGENT INTERMEDIARY

The task of establishing a central control and coordination authority while maintaining local paths of communication and coordination is difficult to

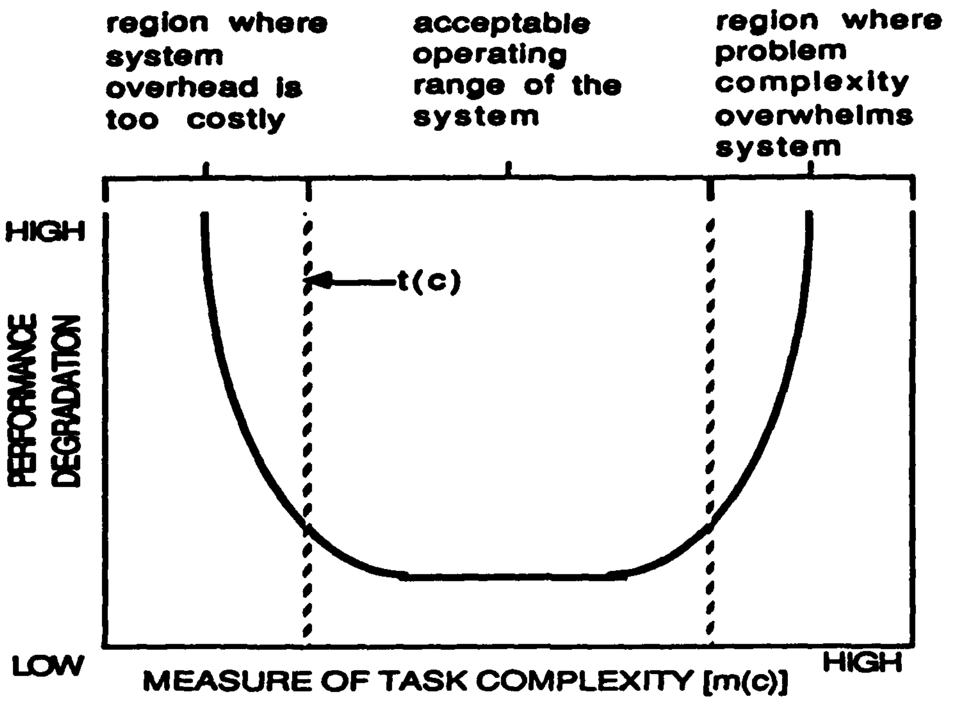


FIGURE 17 - PREDICTED RELATIONSHIP BETWEEN PROBLEM COMPLEXITY, SYSTEM OVERHEAD COSTS, AND PERFORMANCE

accomplish. However, if we want an unobtrusive intelligent intermediary, this is exactly *what must be done*. The system designer must assure that this intelligent intermediary is impartial so as not to interject a bias into the decisionmaking process. In order to do this, the functions and the decisionmaking rules of this entity must be well defined and done. The system designer must assure that this intelligent intermediary is impartial so as not to interject a bias into the decisionmaking process. In order to do this, the functions and the decisionmaking rules of this entity must be well defined and understood. We want the intelligent intermediary to improve performance by facilitating communication, coordination and control within the network without directly working on the problem at hand, but rather by altering the decisionmaking process. Throughout the prior discussion many allusions have been made to some possible functions of the intelligent intermediary. In this section, the previous ideas will be collected and expanded upon in a more focused discussion of the role of the intelligent intermediary in a distributed problem solving network.

The intelligent intermediary is functionally depicted in Figure 18. The intelligent intermediary in general does not observe the environment. The only knowledge it has is the agents' profiles, the group's goals and values, the preferences and responsibilities of the agents, and the agents' evaluation of the information content of messages sent within the network. The intelligent intermediary must also be allowed access to the physical measure of the limited resources in order to calculate a performance index. The intelligent intermediary can periodically poll the responsible agents to learn the current status of the limited resources. The system, on the other hand, will continually monitor the limited resources in order to generate a continuous performance index for data collection purposes. Figure 19 shows some examples of information that can be collected by the intelligent intermediary. This represents a wealth of information that can be used to improve group performance.

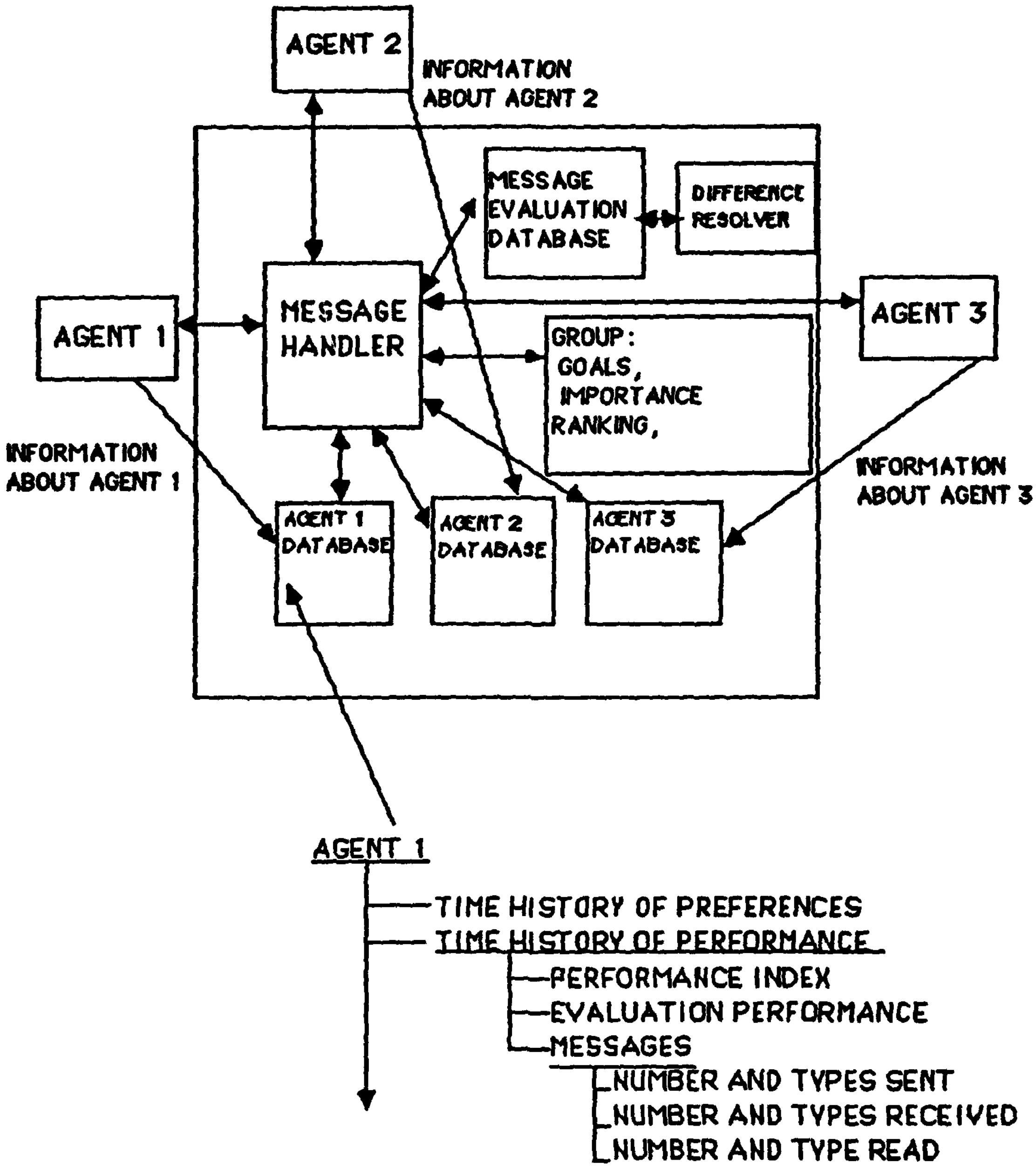


FIGURE 18 - FUNCTIONAL REPRESENTATION OF THE INTELLIGENT INTERMEDIARY

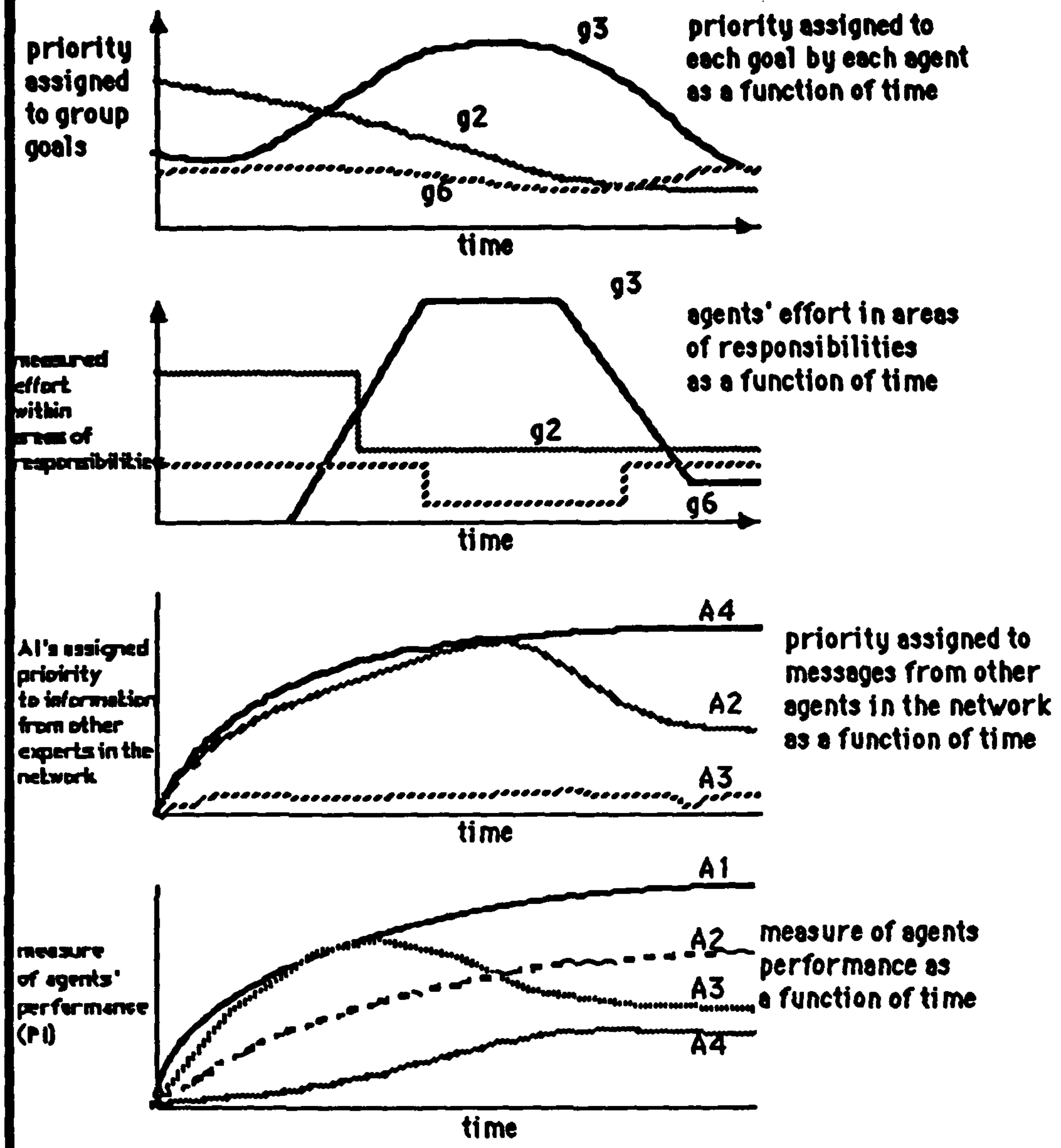


FIGURE 19 - DATA COLLECTED BY THE SYSTEM AVAILABLE FOR USE BY THE INTELLIGENT INTERMEDIARY

One question that can be asked at this point is why limit the functions of the intelligent intermediary? The intelligent intermediary is meant to collect data and to be a communications and learning facilitator that does not directly address the specifics of the problem to be solved. A natural extension of the intelligent intermediary would be an expert system with its own set of decision making rules that could exploit the information collected by the intelligent intermediary. The expert system could interact directly with the other agents in the network to solve the problem. The design of such a system is beyond the scope of this report.

The major tasks of the intelligent intermediary are to monitor the problem definition phase, monitor the distribution of tasks, and to guide the agents' subsequent problem solving efforts. The mechanisms for coordination and control are established during the problem definition and goal setting phase of the problem solving process. The intelligent intermediary will monitor this phase of the problem solving process in order to record group goals, importance rankings, and value assessments. After the problem is stated in terms of an overall global goal which is then decomposed through the definition of layers of sub-goals which support the global goal, then tasks can be distributed throughout the network by assigning specific goals to specific agents. The initial distribution of tasks can be done arbitrarily or based on known areas of expertise of the agents. Subsequent task distribution can be done through mechanisms such as the contract net. The intelligent intermediary must monitor this process to gain knowledge of each agents areas of responsibility.

By assigning a specific goal to a specific agent, control of an agent's efforts can be accomplished. A performance index can be established for the global goal and each sub-goal. The intelligent intermediary can use the performance index to induce agents to perform well, that is control them. If an agent knows his performance is being monitored by an impartial observer under ground rules he has helped to establish and to which he has agreed, peer pressure can be brought upon the incongruous agent. This can be done simply by decomposing the global performance index and distribute each part of the PI based on the assigned areas of responsibility. Each agent would have his own performance index by which the other agents can judge him and on which his reward is based. In a worst case scenario, an agent may be relieved of some areas of responsibilities by the other agents if he were faltering badly. Another major function of the intelligent intermediary is to coordinate information flow within the network. It has to make sure an agent gets the information he requires when he requires it. We want the system to reach a point where the agents spend as little time trying to locate information held by another agent and spends more time executing his tasks. This is a fairly easy task for the intelligent intermediary since it knows the agents' preferences, the agents' areas of responsibility, and the information content of the messages. The task is a simple matter of matching features common to both an agent and a message. Once the initial uncertainty is overcome and the system evolves to a state where the agents have confidence in the intelligent intermediary and confidence in or sure knowledge of the other agents in the system, this task becomes automatic for the intelligent intermediary and second nature for the agents. Establishing a high state of certainty within the network is the most difficult thing to do. However due to the reward system mentioned above, each agent has incentive to gather and correctly evaluate as much information as possible and to disseminate needed information as quickly as possible. The agents also have incentive to process as many incoming messages as possible in the most efficient manner possible. Each agent would have incentive to accurately define his preferences in order to increase his chances of

getting the information he needs to execute his tasks correctly thus garnering more rewards. The agents would have incentive to actively and seriously use the difference resolving feature of the system to expedite the streamlining of communications within the network. In the beginning the overhead costs of this process will probably be significant.The potential benefits of this process is the agents learn about each other and as this happens the number and degree of differences will decrease, the time needed to resolve differences will decrease, and the number of differences that converge will increase. If all of the above does occur, the long term benefits may exceed the start up costs.

5.0 DATA COLLECTION USING THE HYPOTHETICAL TESTBED

Until now, the discussion has focused upon the mechanisms and processes involved in improving group performance in a distributed problem solving network. Now that the testbed has been laid out and the reader has a top level understanding of the testbed, the issues of collecting data and generating meaningful results must be addressed. The standard approach of comparing group performance against that of an optimal controller could be used; however, the tack taken by Adelman et al (1986) is more attractive to the authors.

Adelman et al have adopted a framework in which the process can be analyzed by studying the input/output relationship. Specifically, they investigate the process of cooperation among agents in a distributed problem solving environment. This process was analyzed by monitoring group performance and defined acts of cooperation (outputs) as the incentive structure, the control structure, and the cognitive information (inputs) were systematically varied. The first results that could be expected from the proposed testbed is data showing trends that can be used to understand the basic processes in the distributed problem solving network. Once this is accomplished, we can graduate to the level of Adelman and his associates and try to establish significant relationships. This report will end by proposing candidate processes, accompanying input/output relationships, and plausible trends that would be validated or invalidated by the testbed data.

The processes of interest can be grouped under two broad categories. The first group is cooperation processes (similar to Adelman et al). The natural first step would be to replicate some of results of Adelman et al to exercise the testbed and to establish a baseline. The same inputs can be varied and analogous outputs recorded. Once the baseline is established, the cooperation process can be further investigated by recording how the testbed outputs change as the proposed mechanisms are brought on-line. The second group of interest is uncertainty reduction processes, particularly

the inter-agent uncertainty. The most interesting trend to monitor in the system is the evolution of the communication mode from information broadcasting to selective message sending. If a system blackboard is used to broadcast messages, the individual agent must expend more effort in searching out the information he needs. The opportunities for information being missed are increased due to the increased number of messages proliferating in the system. The communication load is increased as unnecessary messages are being sent or wrong solution paths are followed. The agent's processing load is increased as he has to sort through messages that are unimportant or unrelated to his tasks. The sooner directed messages can be sent, the sooner the problems can be alleviated. With selective messages, only the agents thought to need or to have shown interest in the particular subject content of the message need get the message, instead of all the agents indiscriminately getting the message. The number of messages on the communication lines should correspondingly decrease. The agent has to spend less time worrying about getting the information he needs and can concentrate on his local tasks. This system can only work if the network will evolve to the point where each agent has reasonable confidence that he will get the information he requires without having to continuously and actively search the network for the existence of that piece of information.

The evolution from broadcast to selective message sending is dependent upon reduction of uncertainty in the network, so observing this process is the same.as observing the uncertainty reduction process. In the beginning, almost all of the messages in a simple distributed problem solving network would probably be general queries broadcast to the system blackboard (everybody gets the message) as the agents try to see who is out there and how they want to go about solving the problem. As the problem solving process progresses, the agents will interact with each other and learn about each other and start sending more directed messages to selected members of the network. As a consequence, one expects a smaller number of messages sent. Also,one expects the type of messages being sent on the network to

change from those being questioning in nature to those being informational in nature. The major goal of the proposed system is to accelerate this evolutionary process. The difference resolving rules mentioned above are hypothesized to be a catalyst for this evolutionary process.

Measurable, recordable outputs must be identified. The scenario performance index, PI_i, alone is not a sufficient output measure to evaluate the effectiveness of the proposed system. The overall effectiveness of the categorization system must at least be measured as a function of the following outputs:

- time to decision;
- number and type of messages sent;
- number and type of messages read;
- elapsed time between the sending and reading of a message;
- quality of decision (PI from above);
- number of system interventions;
- difference vectors.

The mechanisms described above are the primary input parameters that would be varied. For example, one expects the different difference resolving rules to have widely varying effects on the process. The degree to which the intelligent intermediary is empowered to intervene may also effect the process. Other interesting inputs to investigate would be the control structures, communications protocols, reward structures, data representation, etc.

The proposed testbed would be a tool used to gather the input/output information needed to analyze the process in question. By analyzing the resultant data, the dynamics of distributed group problem solving might be better understood and new design guidelines established. Once this is done, the design of a system to effect the desired results can begin. The specific outcome is not the paramount object at this

point. If the performance index can be maintained or increased while reducing the time required to make a decision, reducing the number of times the system has to intervene, reducing the time between the sending and reading of a message, increasing the number of certain types of messages sent while reducing the number of other types of messages being sent, one could claim the system works toward reducing traffic on the communication lines and facilitating the decisionmaking process, but we must understand why this is happening. Even if group performance was lowered valuable information can be gained if we can understand what caused the performance loss. Understanding the basic process of the distributed problem solving network is the objective.

6.0 CONCLUSIONS

The testbed at present is envisioned as a useful tool for gathering information about problem solving in a distributed network. The implementation of a full scale system in a real time network might prove too costly in terms of communication and processing load for the capabilities today's technology. Also, the testbed is highly centralized so the benefits of local communications have been lost. The real payoff of such a system in the near term is most probably in a highly structured training arena. By using a controllable, tailored testbed similar to the one described above, a highly efficient team might be built quickly to handle particular problems and its tasks.

The importance of this capability was shown in the SDC studies of the fifties and the case study of the Delta rocket launch team. In both cases, performance increased as people became familiar with each other and then performance started to erode as team members left and had to be replaced by new, inexperienced, and unknown individuals. A certain aspect of human nature arises in group situations whether the group is an informal social group, a professional sports team, or a structured work group: Integrating a new individual, no matter how talented or cooperative, is seldom a smooth transition process. The new member is lacking experience and he does not know the group norms or idiosyncrasies. One method to introduce a new team member would be having the group work on a hypothetical, no risk problem that is unfamiliar to the group. The new member's experience handicap is mitigated and he can interact with the group in a no pressure situation. Each individual has a chance to observe the skills and styles of the others. Data gathered by the testbed/trainer can be used to debrief the agents. The data can point out areas of weakness, areas of strengths, common errors, biases, tendencies, et cetera. Accelerating the assimilation of a new member into the group is a worthwhile endeavor to pursue. In the course of building teams, information can be gathered to increase the design database of such systems and bring a real time implementation of such systems closer to fruition. Another

practical application of such a system may be as a mechanism to gather information for and to tune expert systems. Each agent in the network will be an expert who is given the same problem to solve as the other experts in the network. After each expert offers a solution, the difference resolving function can be used to arrive at a single mutually acceptable solution and the line of reasoning that led to the final answer. This solution and set of decision rules can be compared to the solution and set of decision rules residing in an existing expert system. Any discrepancies between the two can be eliminated by further difference resolving. If the problem is new to the expert system the new information can be entered into the expert system. In turn, such an expert system can be used to train more individuals.

The trainer and expert system tuner are dreams of the longer term future. The promise of the proposed testbed in the near future is to provide a vehicle to gather data needed to understand the basic processes of group problem solving in a distributed network. The processes will be analyzed by investigating their input/output relationships, much in the same manner as classical control theory transfer function analysis. The specific features designed to facilitate the problem solving process can be the varying set of inputs. The output would be the vast database from which the intelligent intermediary draws its information, and thus its power. However, the intelligent intermediary is really an interim measure used to aid the agents until they can learn about each other. As the inter-agent uncertainty is reduced, the role of the intelligent intermediary transitions from that of an active player to that of a vigilant monitor. The reduction of inter-agent uncertainty is a critical process to understand. The set of difference resolving rules used to null out disagreements among the agents is the input to the process and the critical design consideration. The group performance measure is the output of the process and the standard used to judge the effectiveness of the difference resolving rules. Many of the features designed for the implementation of the intelligent intermediary will also serve as data collection mechanisms. Since the

testbed and problem solving facilitator designs exist only on paper and in the mind, results must await further development effort.

7.0 REFERENCES

Adelman, L.; D. A. Zirk; P. E. Lehner; R. J. Moffet and R. Hall: Distributed Tactical Decisionmaking: Conceptual Framework and Empirical Results, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-16, No. 6, November/December 1986, pp. 794-805

Anderson, J. R.: Cognitive Psychology, Artificial Intelligence, Vol. 23, No. 1, May 1984, pp. 1-11

Andriole, S. J.: The Design of Microcomputer-Based Personal Decision-Aiding Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-12, No. 4, July/August 1982, pp. 463-469

Ben-Bassat, M. and A. Freedy: Knowledge Requirements and Management in Expert Decision Support Systems for (Military) Situation Assessment, A Report to the U. S. Army Research Institute for the Behavioral and Social Sciences (AO-A135519), August 1983

Chandrasekaran, B.: Natural and Social Metaphors for Distributed Problem Solving, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 1-5

Chandrasekaran, B.: Distributed Knowledge Base Systems for Diagnosis and Information Retrieval, Report to the Air Force Office of Scientific Research (AFOSR-TR-84-0864), August 1984

Cohen, M. D.; G. Huber; R. L. Keeney; A. H. Levis; L.L. Lopes; A. P. Sage; S. Sen; A. B. Whitson; R. L. Winkler; D. Von Wintersfeld and L. Zadeh: Research Needs and the Phenomena of Decisionmaking and Operations, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 6, November/December 1985, pp. 764-775

Decker, K. S.: Distributed Problem Solving Techniques: A Survey, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-17, No. 5, September/October 1987, pp. 729-739

Farrel, R. L.; S. Bonder; L. D. Proeger; G. Miller and D. E. Thompson: Capturing Expertise: Some Approaches to Modeling Command Decisionmaking in Combat Analysis, IEEE Transactions onSystems, Man, and Cybernetics, Vol. SMC-16, No. 6, November/December 1986, pp. 766-773

Fox, M. S.: An Organizational View of Distributed Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 70-80

Hampton, D. R.: Contemporary Management Theory, McGraw-Hill Book Company, New York, 1977

Hayes-Roth, B.: A Blackboard Architecture for Control, Artificial Intelligence, Vol. 26, No. 3, July 1985, pp. 251-321

REFERENCES (continued)

Keeney R. L. and Raiffa H.: Decisions with Multiple Objectives: Preferences and Value Tradeoffs, John Wiley and Sons, New York, 1976

Knaeuper, A. and W. B. Rouse: A Rule-Based Model of Human Problem Solving Behavior in Dynamic Environments, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 6, November/December 1985, pp. 708-719

Kolodner, J. L.: Towards an Understanding of the Role of Experience in the Evolution From Novice to Expert, International Journal on Man-Machine Studies, Vol. 19, 1983, pp. 497-518

Lesser, V. R.. and D. D. Corkill: Functionally Accurate, Cooperative Distributed Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 81-95

Lesser, V. R. and D. Corkill: The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks, The AI Magazine, Fall 1983, pp. 15-32

Lesser, V. R.; D. Corkill; J. Pavlin; L. Lefkowitz; E. Hudlika; R. Brooks and S. Reed: A High-Level Simulation Testbed for Cooperative Distributed Problem Solving, Proceedings form the Third International Conference on Distributed Computing Systems, October 1982, pp. 341-349

Luft, J.: Group Processes: an Introduction to Group Dynamics, Mayfield Publishing Company, Palo Alto, Ca, 1970

Moray, N.; P. Loosteen and J. Pajak: Acquisition of Process Control Skills, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-16, No. 4, July/August 1986, pp. 497-504

Morris, N. M. and W. B. Rouse: The Effects of Type of Knowledge upon Human Problem Solving in a Process Control Task, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 6, November/December 1985, pp. 698-707

Morris, N. M.; W. B. Rouse and J. L. Faith:PLANT: An Experimental Task for the Study of Human Problem Solving in Process Control, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 6, November/December 1985, pp. 792-798

Parsons, H. M.: Man-Machine Systems Experiments, The John Hopkins Press, Baltimore, Maryland, 1972

REFERENCES (continued)

Rubinstein, M. F. and K. Pfeiffer: Concepts in Problem Solving, Prentice-Hall Inc., Englewood Cliffs, New Jersey,1984

Rubinstein, M. F.: Patterns of Problem Solving, Prentice-Hall, Englewood Cliffs, New Jersey, 1975

Sage, A. P.: Behavioral and Organizational Considerations in the Design of Information Systems and Processes for Planning and Decision Support, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 9, September 1981, pp. 640-678

Sage, A. P.: Information Systems Engineering for Distributed Decisionmaking, IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-17, No.6, November/December 1987, pp. 920-936

Sage, A. P. and W. B. Rouse: Aiding the Human Decisionmaker Through the Knowledge-Based Sciences, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-16, No. 4, July/August 1986, pp. 511-521

Schwartz, J. P.; J. H. Kullback and S. Schrier: A Framework for Task Cooperation within Systems Containing Intelligent Components, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-16, No. 6, November/December 1986, pp. 788-793

Smith, R. G. and R. Davis: Frameworks for Cooperation in Distributed Problem Solving, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 61-70

Sondheimer, N. K. and N. Relles: Human Factors and User Assistance in Interactive Computing Systems: An Introduction, IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-12, No. 2, March/April 1982, pp. 102-107

Stephanou, H. and A. P. Sage: Perspectives on Imperfect Information Processing, IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-17, No. 5, September/October 1987, pp. 780-798

Szilagyi, A. D. and M. J. Wallace: Organizational Behavior and Performance, Goodyear Publishing Company, Santa Monica California, 1980

Wang, M. S. and J. F. Courtney: A Conceptual Architecture for Generalized Decision Support System Software, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-14, No. 5, September/October 1984, pp. 701-711

REFERENCES (continued)

Wesson, R.; F. Hayes-Roth; J. W. Burge; C. Stasz and C. A. Sunshine: Network Structure for Distributed Situation Assessment, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 5-23

Witus, G.: Decision Support for Planning and Resource Allocation in Hierarchical Organizations' IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-16, No. 6, November/December 1986, pp. 927-942

Yang, J.D.; M. N. Huhns and L. M. Stephens: An Architecture for Control and Communications in Distributed Artificial Intelligence Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 3, May/June 1985, pp. 316-326

Interim Report
Distributed Problem Solving: Adaptive Networks with a Computer Intermediary Resource.

Focus Topic:
The Distributed Situation Assessment Problem---
Instructions and Programs

Principal Investigator:
John Lyman

Man-Machine-Environment Engineering Laboratory
Department of Material Science & Engineering
UCLA
Los Angeles, CA 90024
(213) 825-8486

June 29, 1988

Prepared by:
Steven A. White
John Lyman

Contract #MDA903-84-C-0355
Army Institute for Research

Contract Monitor:
Mike Drillings

Contractor:
The Regents of the University of California
University of California, Los Angeles
Office of Contracts and Grants Administration
405 Hilgard Ave
Los Angeles, CA 90024

The views, opinions, and findings contained in this report are those of the authors and should not be construed as an official Department of Army position, policy, or decision, unless so designated by other official documentation.

The Man-Machine-Environment Engineering Laboratory

The Distributed Situation Assessment Problem*

Overview and General Instructions

There are two separate programs that are divided among four Mac Pluses. The four computers are designated thus:

Mac0: The Experimenters station--uses program MacZero.

Mac1: Subject A's station--uses program Distrib.

Mac2: Subject B's station--uses program Distrib.

Mac3: Subject C's station--uses program Distrib.

The four computers must be connected on an AppleTalk Network. The programs read the Chooser Name of their computer. Therefore, before running any of the programs, the Chooser DA must be selected [illegible] name must be entered which corresponds to that computer's station: The Chooser Names are Mac0, Mac1, Mac2, or Mac3.

**The programming for this research was done with MacApp Pascal by Stephen A. White of the Man-Machine-Environment Engineering Laboratory. The MPW Assembly sub-routines to drive the AppleTalk communications were adapted from a program named MegaTalk that was developed at MegaGraphics Inc. Appreciation is expressed to Mr. Chris Hull for his cooperation in supplying the source code for the MegaTalk program.*

Make sure all four computers have started their programs before attempting to send anything between them. At start-up, each programs registers itself on the network. When a program sends its first message, it checks who is on the network and makes a table of names. Each program only does this once, so if one of the other computers is not registered when the table is made, it will not be recognized in later communications.

When each of the three subject stations have booted up the program (Water), the Debug window first comes up with some programmer information, then the Welcome Screen comes up:

Welcome to the Distributed

Problem Solving Experiments

Figure 1. Welcome Screen.

At this point the experimenter (or trained subject) presses the mouse key once and the Group Number Dialog Box appears. The experimenter enters the appropriate number and clicks on 'Finished'. The group number is stored in each computer and is sent to Mac0.

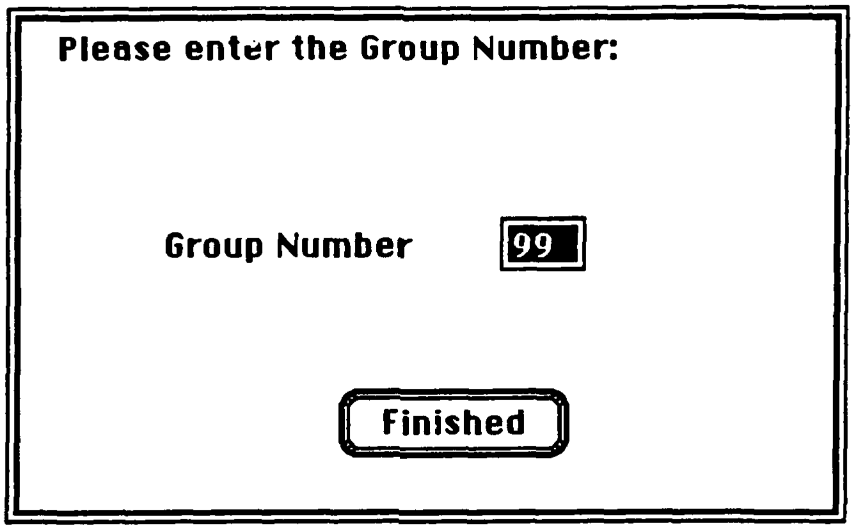


Figure 2. Group Number Dialog Box.

At the same time at the experimenter's station, the program (WaterMacZero) has been booted up and has reached the Welcome Screen. The experimenter presses the mouse key once and then the Task Parameter Dialog Box appears. After each subject station has sent their group number to Mac0, the experimenter selects the appropriate parameters for the current scenario and clicks on 'Finished'.

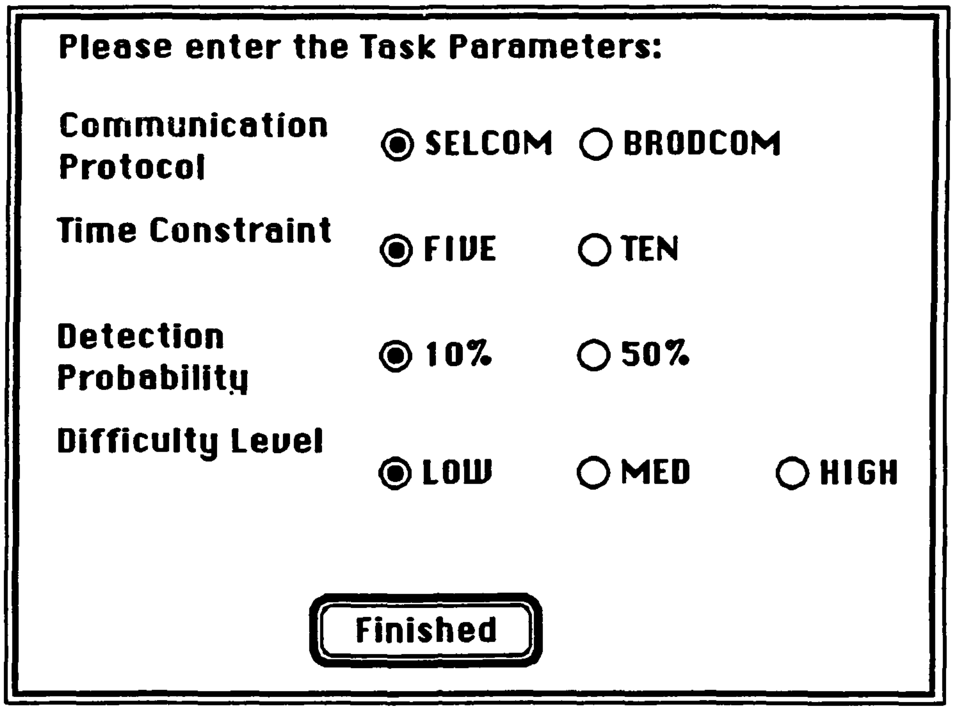


Figure 3. Task Parameter Dialog Box.

The task parameters are then sent to each of the three subject stations.

When the task parameters have been received by the subject stations it triggers a series of Instruction Screens that the subjects read. There is a 'Next' button at the bottom of each screen that the subject clicks on after he or she is done reading that screen. The experimenter is also explaining the scenario and answering any questions at this time.

The last screen asks the subjects if they are ready and they click on the 'Ready' button when they are. When they press the 'Ready' button their station sends a message to Mac0 that they are ready. When Mac0 has received a ready message from all three of the subjects, Mac0 sends a message to each of the subject stations to begin the scenario.

The scenario begins:

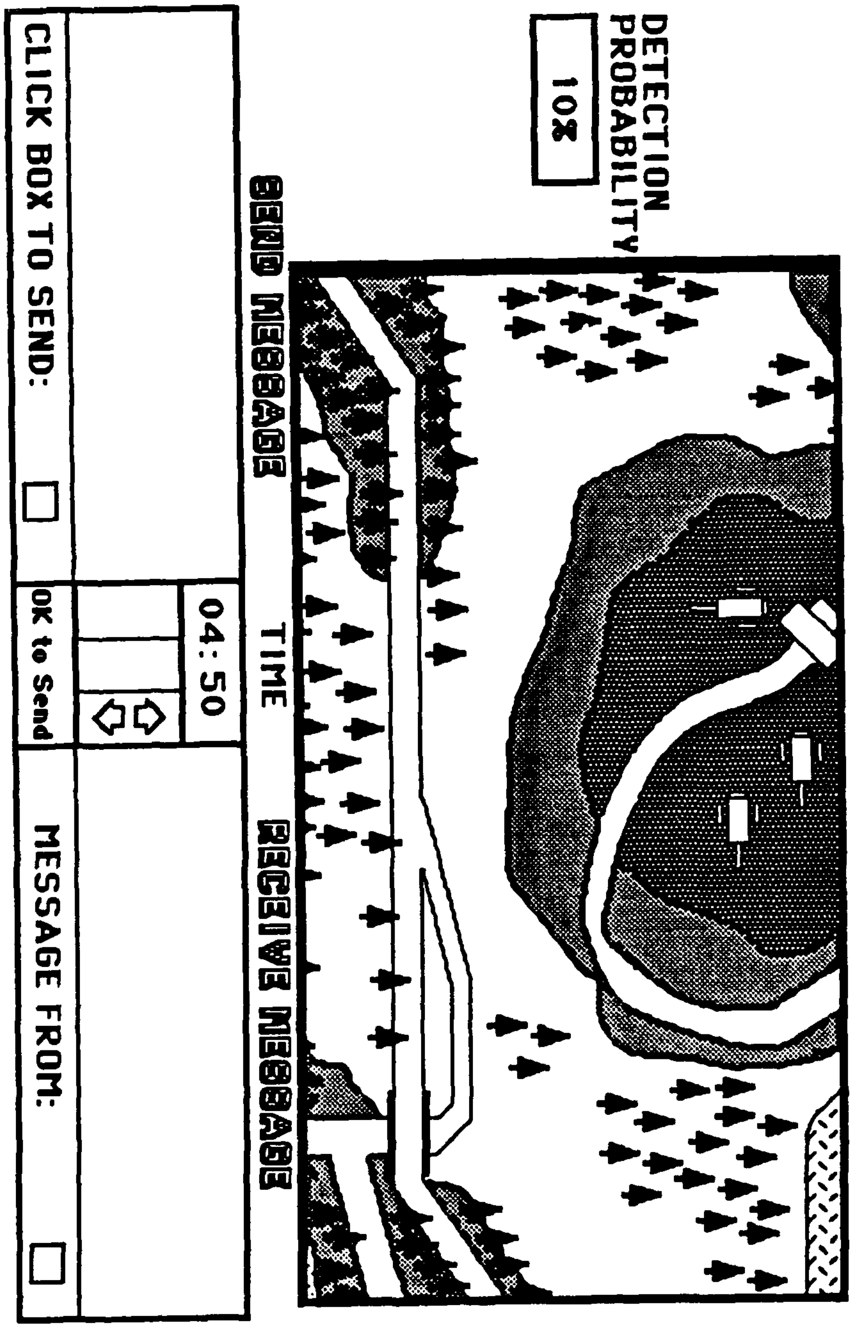


Figure 4. Scenario Screen

The scenario screen is made up of many elements, some of which are under the control of the subjects and some are not. The menu bar is used in the program but there are three Command Keys available (mostly for programming and debugging). ⌘D brings the Debug Window forward which displays the writeln messages inserted by the programmer. ⌘E ends the scenario early (used by the programmer). ⌘Q quits the program.

Because each subject only sees a subset of the scenario, communication between subjects is an important factor for the group to achieve its goal. The content of the messages generally will consist of informing the other subjects what part of the map that subject see, including landmarks such as rivers, lakes, bridges and buildings. Any moving object on the screen (tanks or jets) are the enemy and subjects will communicate the presence and position of these objects. The goal of the group is to correctly assess the situation involved in the scenario. This includes determining their physical relationships to each other, how many enemy elements there is, and the objective of the enemy.

When a subject begins typing the message is displayed in the 'Send Message' box. In the Broadcast Scenario, to send the message, the subject clicks on the small box below the message. In the Hierarchical Scenario, to send the message, the subject has to choose which of the other two subjects to send to by clicking on one of the two boxes which have the letter of the appropriate subject within them.

While a message is in the process of being sent, the 'OK to Send' box is filled in and all input is ignored so that an attempt at sending another message is not possible. When the 'OK to Send' is cleared then new input will be accepted and another message can be sent.

Messages arriving at a subject station will be displayed in the 'Receive Message' box. The first message will be displayed and subsequent messages stored in a list. In the box just to the right of the 'OK to Send' box the number of messages that have been received will be displayed. In the box just above the 'OK to Send' box and just to the right of the up and down arrows the number, in the list, of the message currently shown will be displayed. To change the message being shown, the subject clicks on the down-pointing arrow (above the 'OK to

Send' box) to move down the list and see more recent messages. By clicking on the up-pointing arrow, the subject can move up the list to review older messages. A maximum of 15 messages are kept in the list at one time. The first message in will be the first message out when the 16th message is received.

When the scenario is over the subject is informed and then the Scenario Assessment Section of the experiment begins.

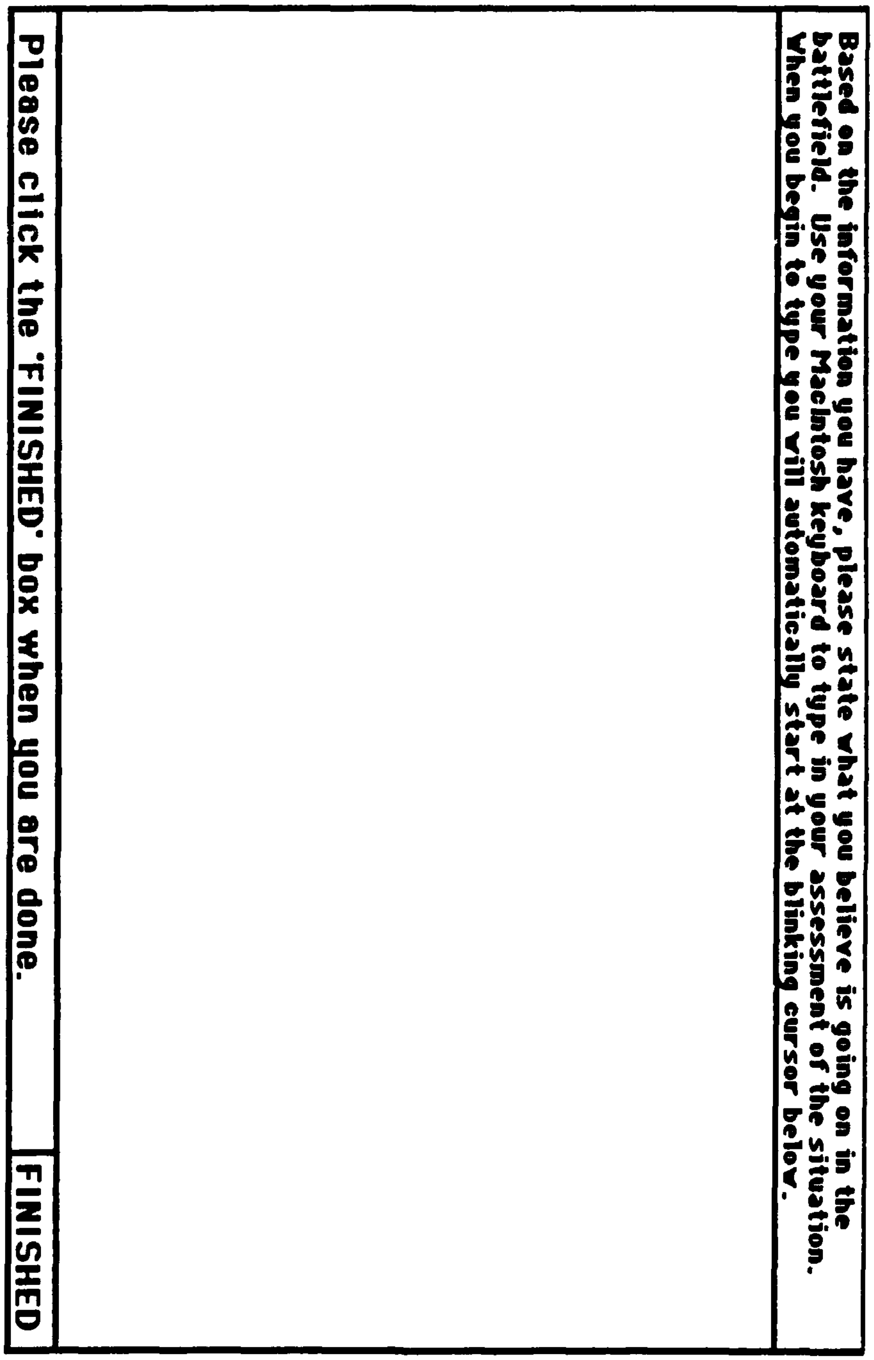


Figure 5. Scenario Assessment Screen.

When the subject is finished with the Scenario Assessment Section the Workload Assessment Section of the experiment begins. The first screen includes the instructions of how the subjects are to enter their workload assessments.

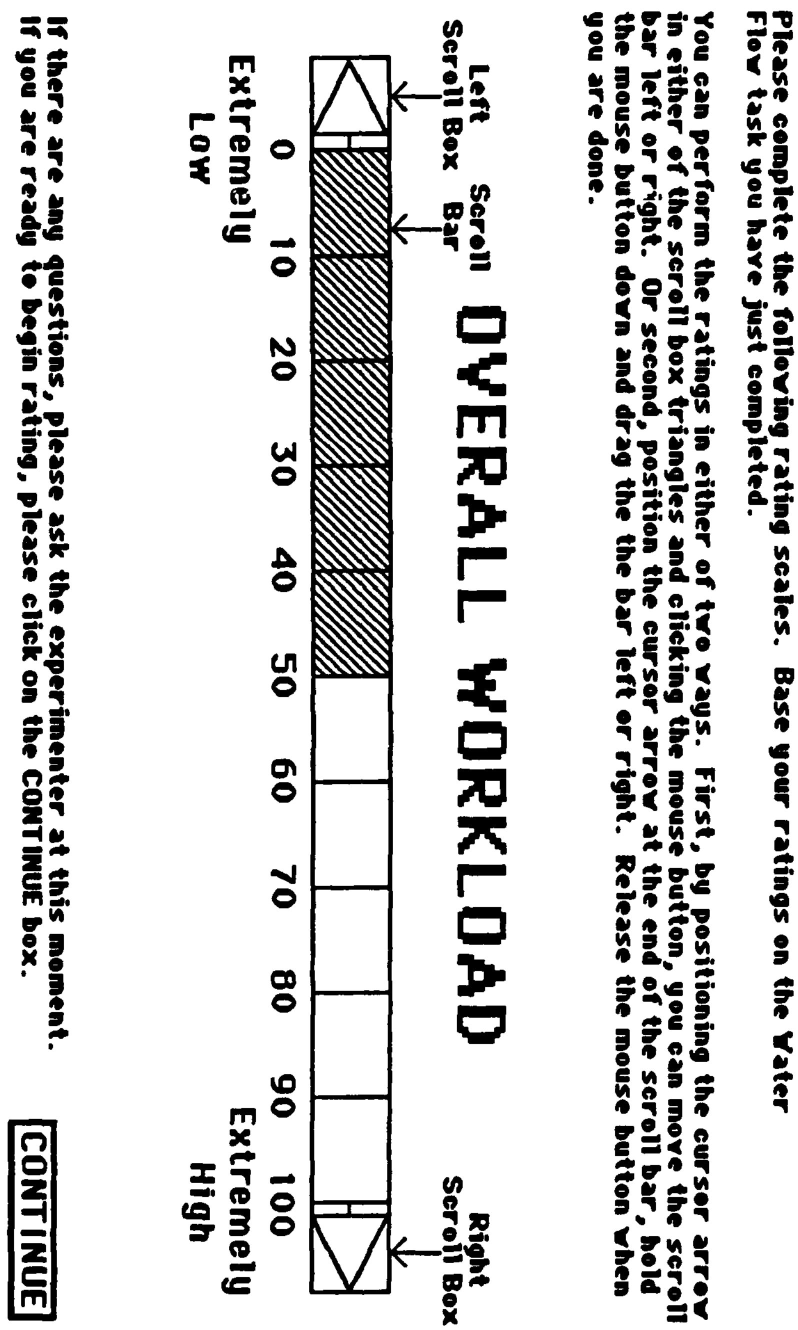


Figure 6. Workload Assessment Instructions.

There are nine scales that range of 0 to 100. The nine scales, which the subjects are to rate to they felt during the task, are: Overall Workload, Task Difficulty, Time Pressure,

Performance, Mental and Sensory Effort, Physical Effort, Frustration Level, Stress Level, and Fatigue. The subjects can adjust the rating scales (each scale starts at 50) either by clicking the mouse in one of the two directional arrows at each end of the scale or by clicking on the edge of the shaded rating box and dragging the edge forward or backward.

After the Workload Assessment section has been completed, the subjects have one more task. In the Assessment Agreement section the subjects read the other subjects' assessments and then must rate whether they agree or disagree with those assessments. Each subject is shown his or her own assessment in the top panel and the other two assessment below. Sometimes there is a delay if one of the subjects is slow on the assessment section or the workload section. Mac0 will not send over the assessments until all the subjects has caught up. The subject click on one of the six rating boxes (labeled 0 to 5) to show their degree of agreement for each of the other assessments. When they make their rating a dialog box appears and asks them to state the reasons why they made the rating that they did.

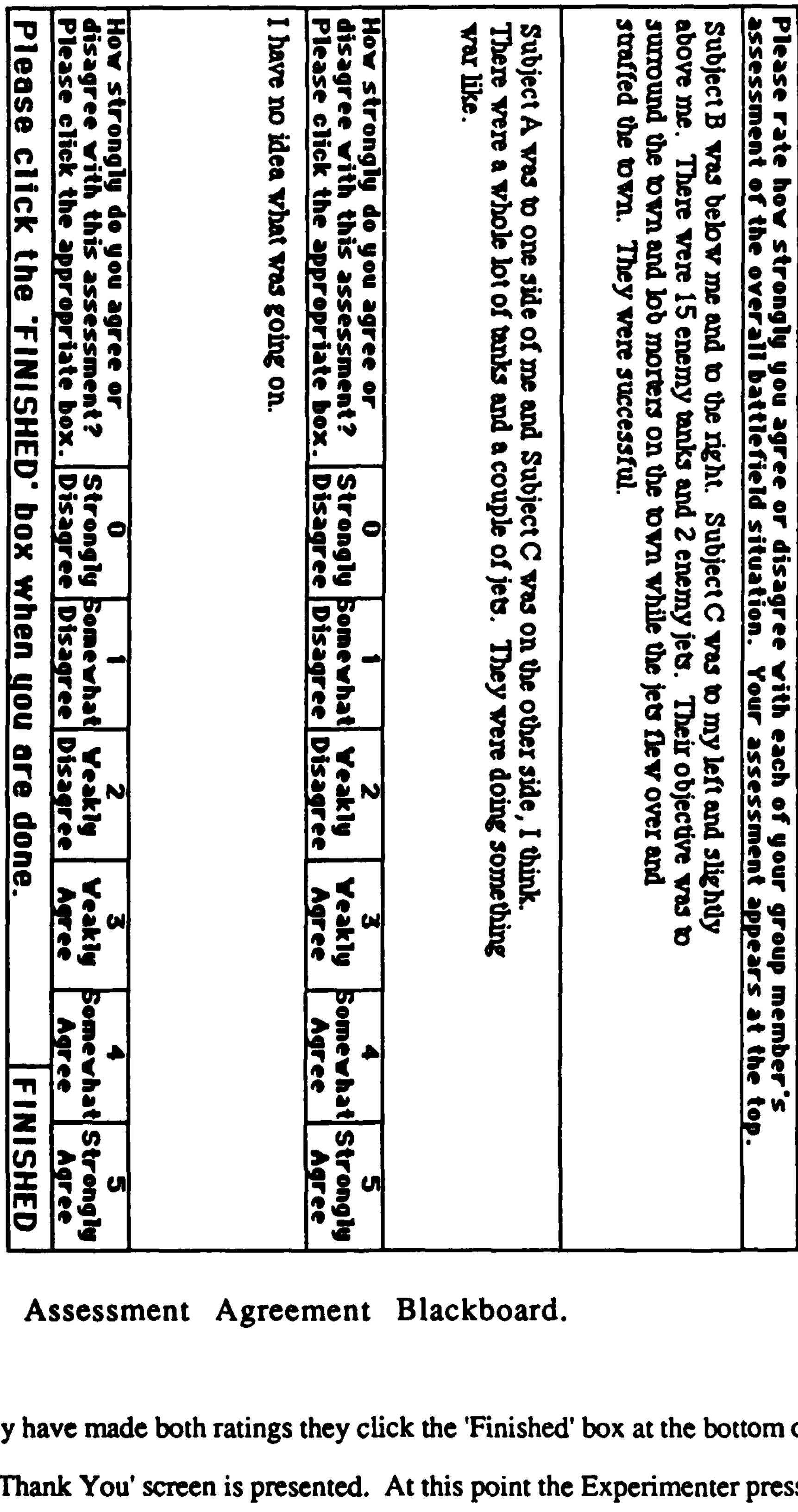


Figure 7. Assessment Agreement Blackboard.

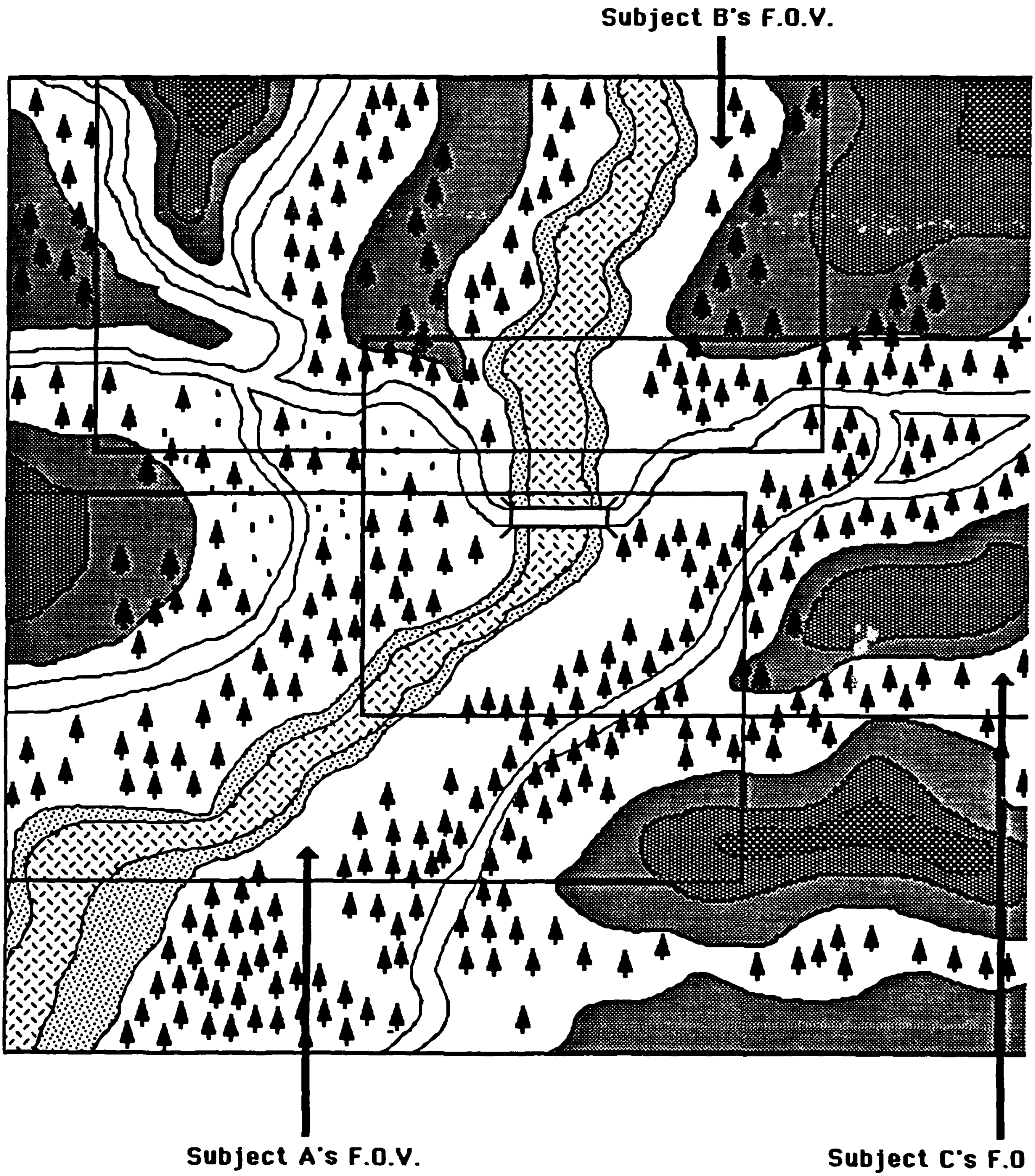
After they have made both ratings they click the 'Finished' box at the bottom of the screen and then the 'Thank You' screen is presented. At this point the Experimenter presses ⌘Q to quit the program and they are done with that scenario. When the Experimenter presses ⌘Q to

quit the Mac0 program three 'Save this Document' Dialog Boxes will appear so that the data accumulated in the scenario can be stored.

Thank You for Participating in the Distributed Problem Solving Experiment

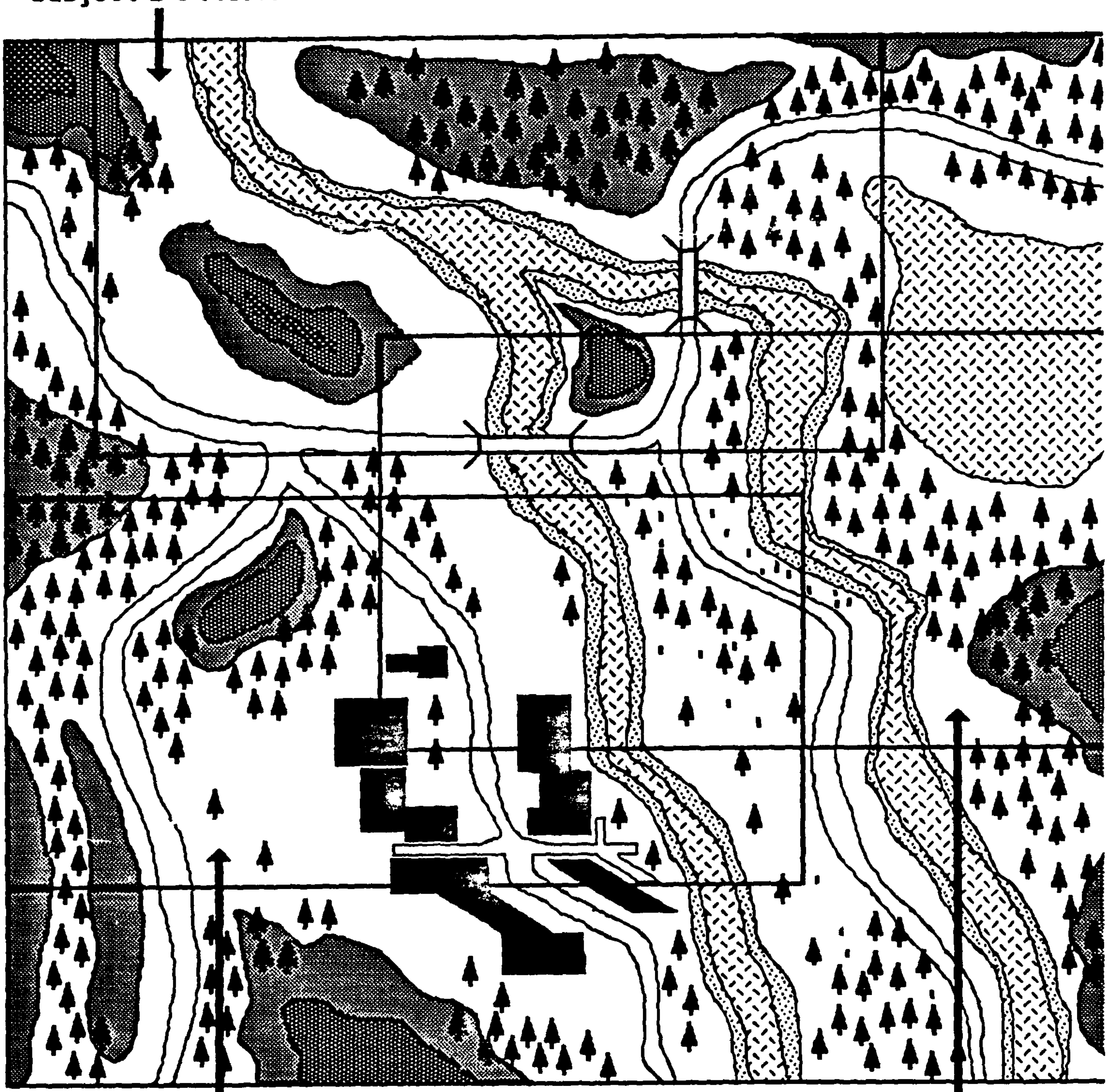
Figure 8. Thank You Screen.

SUBJECTS' FIELD OF VIEW FOR THE LOW-LEVEL COMPLEXITY BATTLEFIELD



SUBJECTS' FIELD OF VIEW FOR THE MID-LEVEL COMPLEXITY BATTLEFIELD

Subject B's F.O.V.



Subject A's F.O.V.

Subject C's F.O.V.

SUBJECTS' FIELD OF VIEW FOR THE HIGH-LEVEL COMPLEXITY BATTLEFIELD